Utilities Plus

Some of the most useful utility programs all combined in this one fantastic package

MichTron®

Utilities Plus

Five Utility Packages

for the Atari ST

Reference Manual

Published By MICHTRON, Inc.

YOUR RIGHTS AND OURS: This copy of **Utilities *Plus*** is licensed to you. You may make copies for your own use or for archival storage. You may also sell your copy without notifying us. However, we retain copyright and other property rights in the program code and documentation. We ask that **Utilities *Plus*** be used either by a single user on one or more computers or on a single computer by one or more users. If you expect several users of **Utilities *Plus*** on several computers, contact us for quantity discounts and site-licensing agreements. Also if you intend to rent this program, or place this program on a BBS, contact us for the appropriate license and fee.

We think this user policy is fair to both you and us; please abide by it. We will not tolerate use or distribution of all or part of **Utilities *Plus*** or its documentation by any other means.

LIMITED WARRANTY: In return for your understanding of our legal rights, we guarantee **Utilities *Plus*** will reliably perform as detailed in this documentation, subject to limitations here described, for a period of thirty days. If **Utilities *Plus*** fails to perform as specified, we will either correct the flaw(s) within 15 working days of notification or let you return **Utilities *Plus*** to the retailer for a full refund of your purchase price. If your retailer does not cooperate, return **Utilities *Plus*** to us. While we can't offer you more cash than we received for the program, we can give you this choice: 1) you may have a cash refund of the wholesale price, or 2) you may have a merchandise credit for the retail price, which you may apply toward buying any of our other software. Naturally, we insist that any copy returned for refund include proof of the date and price of purchase, the original program disk, all packaging and documentation, and be in salable condition.

If the disk on which **Utilities *Plus*** is distributed becomes defective within the warranty period, return it to us for a free replacement. After the warranty period, we will replace any defective program disk for $5.00.

We cannot be responsible for any damage to your equipment, reputation, profit-making ability or mental or physical condition caused by the use (or misuse) of our program.

We cannot guarantee that this program will work with hardware or software not generally available when this program was released, or with special or custom modifications of hardware or software, or with versions of accompanying or required hardware or software other than those specified in the documentation.

Under no circumstances will we be liable for an amount greater than your purchase price.

Please note: Some states do not allow limitations on how long an implied or express warranty lasts, or the exclusion or limitation of incidental or consequential damages, so some of the above limitations or exclusions may not apply to you.

UPGRADES AND REVISIONS: If you return your information card, we will notify you if upgrades to **Utilities *Plus*** become available. For minor upgrades and fixes, return the original disks to us with $5.00. For major revisions, the upgrade fee is typically 15-20% of the original suggested retail price.

FEEDBACK: Customer comments are VERY important to us. We think that the use, warranty and upgrade policies outlined above are among the fairest around. Please let us know how you feel about them.

Many of the program and documentation modifications we make result from customer suggestions. Please tell us how you feel about **Utilities *Plus*** — your ideas could make the next version better for all of us.

Utilities *Plus*

Reference Manual

Published by MichTron, Inc.

United States
576 South Telegraph
Pontiac, Michigan 48053
☎: (313)334-5700
BBS: (313)332-5452

United Kingdom
Box 68 St. Austell
Cornwall, PL25 4YB
☎:0726 68020

88 89 90 91 10 9 8 7 6 5 4 3 2 1

ISBN 0-944500-05-6

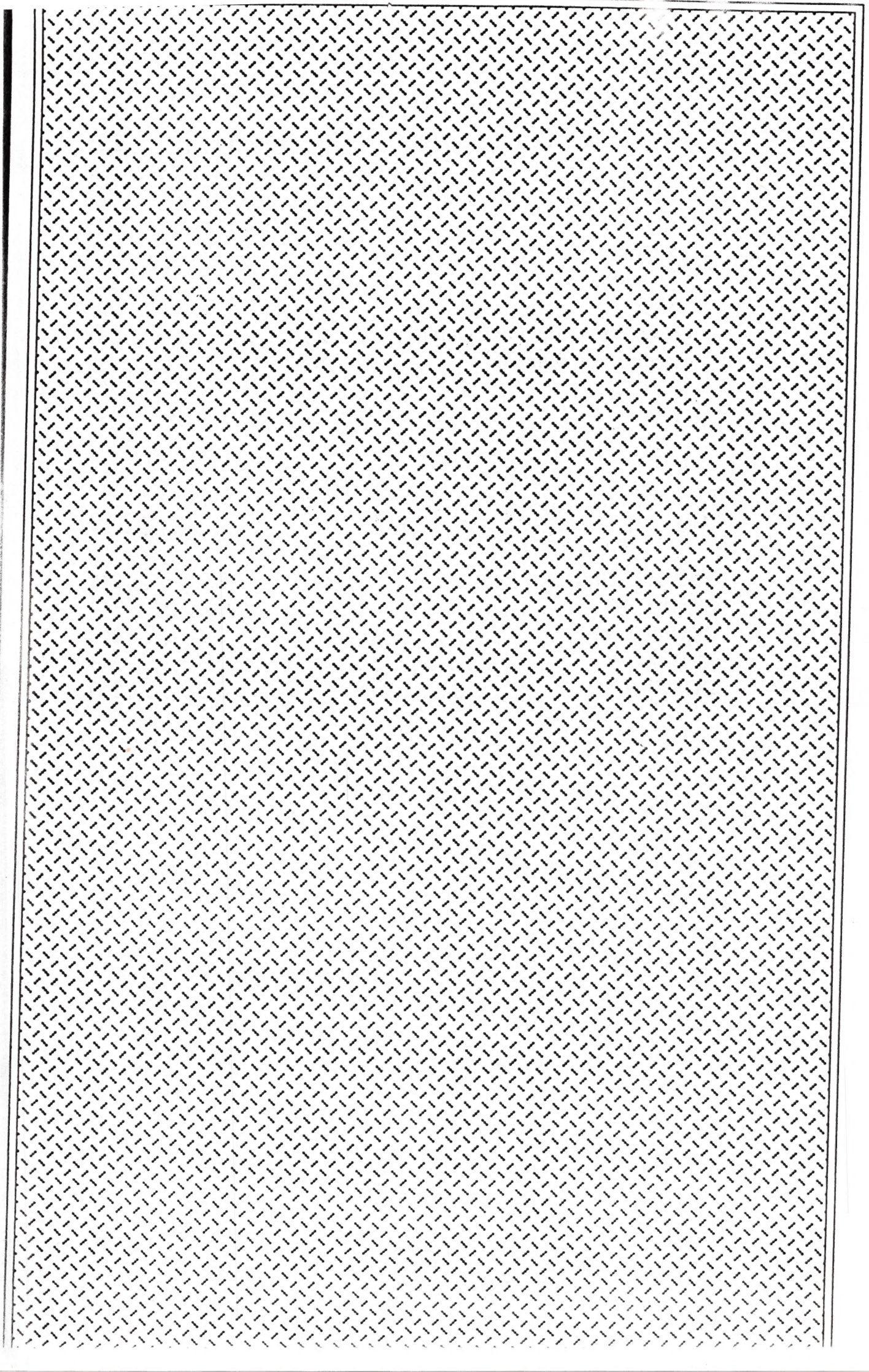
Contents

Part II: STuff

Part III: Superdirectory

Part IV: M-Disk *Plus*

Part V: DOS Shell



An Introduction To

Utilities Plus

Introduction

Utilities *Plus* is a combination of five great packages that have been separately released by MICHTRON. Now they are combined into one dynamic package that will maximize the potential of your *Atari ST*.

MichTron Utilities: A collection of utilities that help maintain control over your *Atari ST* files by reading and changing individual bytes of information.

STuff: Programs and desk accessories designed to minimize the hassles and maximize the benefits of your *ST*.

Superdirectory: A powerful, easy to use disk cataloging program which runs under the GEM operating system. It allows you to keep track of all of your floppy and hard disk files in a convenient format.

M-Disk *Plus*: A combination of **M-Disk** and **Soft-Spool**. These programs reserve a portion of your computer's memory to load and save data and to allow you to work while printing.

DOS Shell: A program that allows you to mimic DOS commands on the GEM-based *Atari ST* for speed and convenience.

General Instructions

Creating a Back UP Copy

Before you use any of the disks you should make a Back Up copy of the disk and store the original in a safe place. The **Utilities *Plus*** disk is not copy-protected to allow you to make a security copy. We request that the buyer not give the program to third parties, for only with the user's integrity will it be possible to bring high quality software into the marketplace at a reasonable price in the future.

Refer to your computer's manual for an explanation of how to copy (back-up) files.

Hardware Requirements

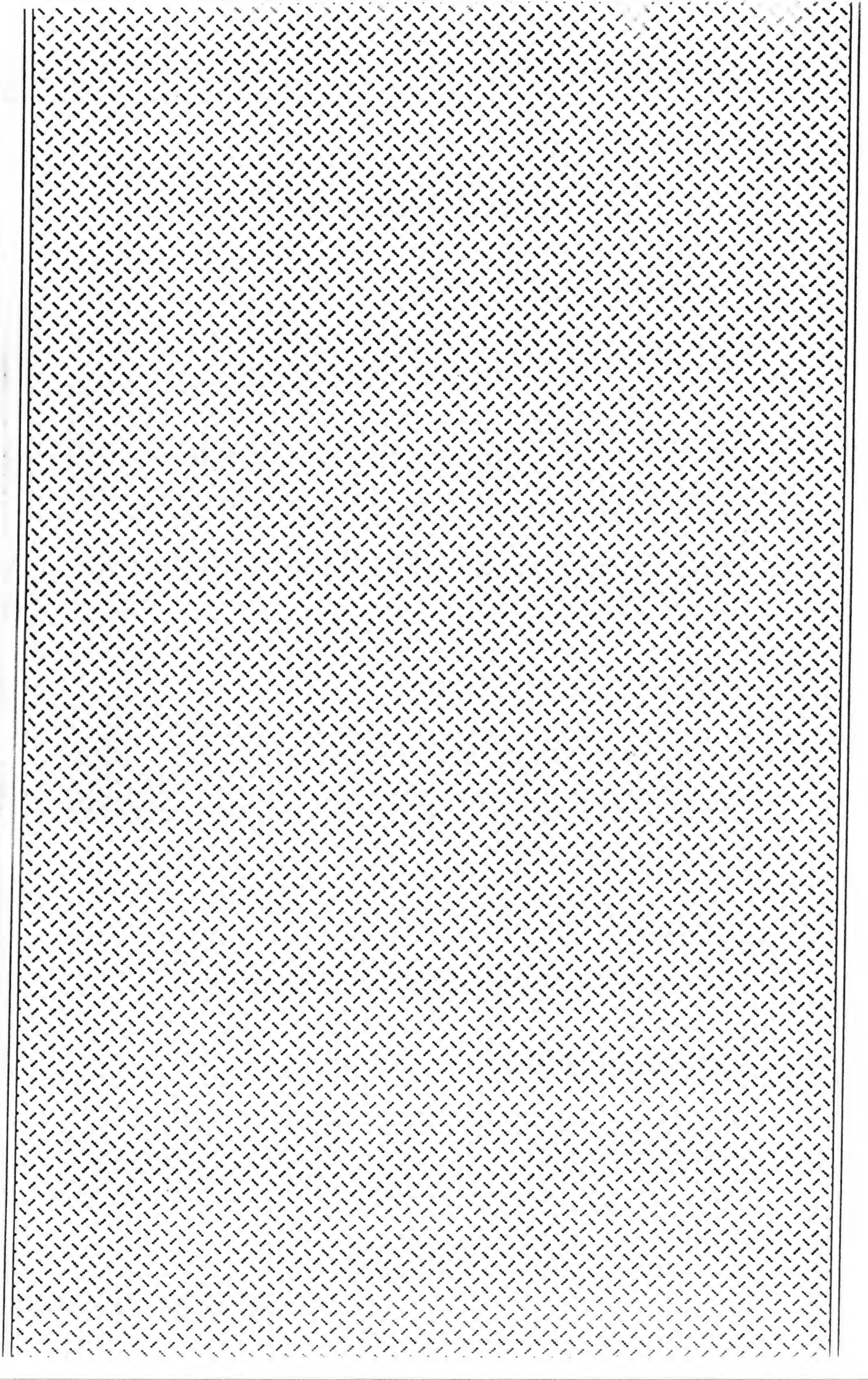
You will need a minimum system of an *Atari ST* with a monochrome monitor for all of these programs. Some programs may have more requirements; see the individual **Introductions** to each section for each program's specific requirements.

Loading

See individual **Introductions** to each section.

READ.ME File

Updates to this documentation (if any) are stored in a file called READ.ME. If this file exists on your disk, double click on the icon and then click on *Show* or *Print* to display the contents. Be sure to correct or revise the relevant portions of this manual.



Part I

MichTron Utilities

Utilities that Maintain Control

Over Atari ST Files

by Reading and Changing

Individual Bytes of Information

By Timothy Purves

Chapter 1:

An Introduction to MichTron Utilities

Introduction

MichTron Utilities helps you maintain control of your *Atari ST* files, even when things go wrong. Using **MichTron Utilities**, you can read and change any byte of information anywhere on your floppy or hard disks.

With this ability to address individual bytes, you can:

- Search through files or disks
- Change file contents
- Change file and volume names
- Change file attributes
- Format individual disk tracks
- Copy or verify individual sectors
- Restore deleted files
- Recover data from damaged disks
- Repair damaged disks

MichTron Utilities offers more flexibility in handling files and has more potential to deal with trouble when it arises.

For instance, you can hunt for a file containing specific information even if you can't remember the file name. You can change the attributes of any file to hide it, make it a system file, make it read-only, change its date, or change its name. You can also change the volume name.

If you accidentally delete a file, you can recover it as long as you haven't physically overwritten it. Even if the file has been partially destroyed, you may be able to recover the remaining parts of it.

If a disk has been damaged, you can isolate the trouble area and recover information from the rest of the disk. You can even try to reformat the affected area without harming the rest of the disk.

— WARNING —

Use Caution with this Program.

Because you can use this program to reach into any disk location and change the information stored there, you can permanently destroy information. It is possible to destroy all the information on a disk with a single command.

If you are an experienced programmer with detailed knowledge of disk file management techniques, you should find it easy to use **MichTron Utilities** without trouble.

If you are new to disk handling, be especially careful. We have provided operation-specific procedures to solve common problems. Please do not try making up your own techniques unless you are willing to risk destroying some information while you learn.

As always, your best protection against data-loss disaster is to back up your disks. Even this program cannot promise to save your data in every situation.

If possible, don't use this program on the only disk you have containing valuable information. If, of course, you bought this package to recover the unduplicated database you spent three months compiling, you'll have to work with the original. But practice first on something else. We want you to be satisfied with the results, not upset because you misread the manual and permanently erased your data!

Before you proceed with any function, make certain the function is the one you want, that the sources and destinations are correctly entered, and that the right disks are in the right drives. Taking a few moments to check everything at every step can prevent a disastrous error.

Hardware Requirements

MichTron Utilities was designed for the *Atari ST* computer, independent of the monitor, printer or other peripherals that may be installed. Only one single- or double-sided disk drive is required.

> ***NOTE:*** *If you have a color monitor, set your preferences for* Medium Resolution *before loading this program. If you try to use* Low Resolution, *an alert will insist that you change.*

Loading Instructions

To load **MichTron Utilities**, boot up your *Atari ST* with your TOS disk or from TOS in read-only memory. Set the resolution preference to Medium or High Resolution.

Next, insert your working copy of **MichTron Utilities** into one of your disk drives. Press ESC if necessary to alert the computer that you have changed disks.

Select the program MUTIL.PRG by double-clicking on its icon or label.

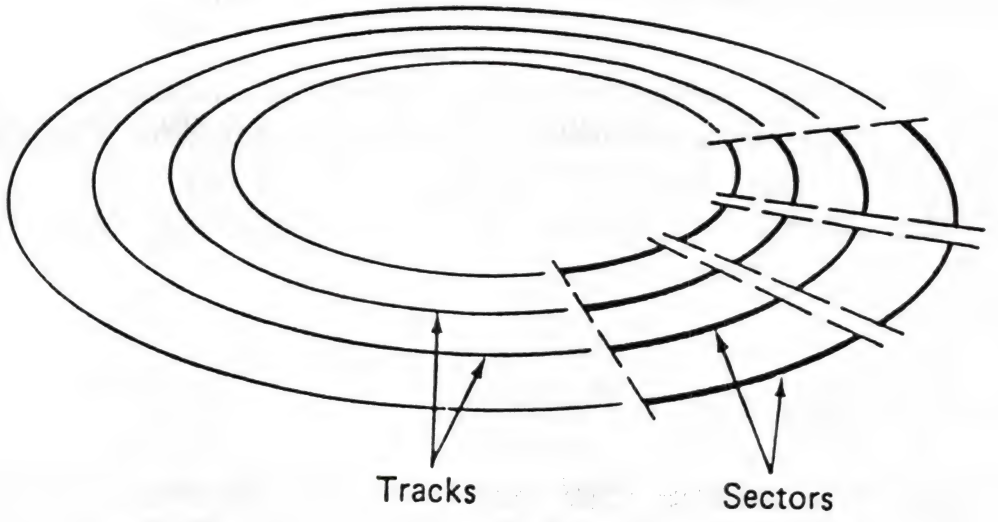
The **MichTron Utilities** opening screen will appear. Pull down an appropriate menu to begin.

How Files are Stored on Disk

In ordinary disk file handling you might never need to know how the operating system stores files on your disks: after all, the operating system is supposed to do the housekeeping for you and let you work on more important things.

But if you want to change file contents or structure or recover damaged or deleted files, it's vital that you know how your files are labeled and where to look for both files and labels.

Each floppy disk contains 80 circular, concentric tracks numbered from 0 on the outside edge to 79 on the inside. Each track is divided into nine sectors on each side, numbered 1 to 9 on the first side, 10 to 18 on the second side if there is one. Single-sided disks have 720 sectors in 80 tracks, double-sided disks 1440 sectors in 80 tracks. Each sector contains 512 bytes of information.



The logical sequence of sectors begins with sector one on track zero, then counts through all nine or eighteen sectors on track zero. The next sector in logical order is sector one on track one. The sequence continues through all sectors on each track before advancing to the next track.

The first few sectors on every disk are reserved for special use by the operating system. All other sectors are available for general use, so it is in those other sectors that we will find our files stored.

But those first few sectors contain all the information we need to find our files on the disk. Therefore both portions of the disk are important.

Hard disks allot storage similarly, but the disk may be logically subdivided by an operating system into several segments that operate independently.

The Boot Record

The first sector on any disk is loaded with a short machine-language program designed to load the operating system into memory from the disk. Many disks, of course, do not have the operating system on them, but the loading program is always present anyway.

The File Allocation Table

The next few sectors (the exact number depends on the type of disk being used) contain a list of how all the sectors on the disk are being used. The data sectors are grouped together into clusters of one to a few sectors each (on any disk, all clusters are the same size, but different storage media use different cluster sizes).

When the computer stores files on the disk, it looks for unused clusters and writes the first part of your file into the first one it finds. If more space is needed, another cluster will be chosen to store the next part. The second cluster need not be physically near the first. The whole purpose of the file allocation table (FAT) is to keep track of where the clusters belonging to a file are located.

Each cluster in use by any file is tagged in the FAT with the number of the next cluster used by the same file. If the cluster is the last used by the file, a special end-of-file mark appears in the FAT. Other special tags show that a cluster is not now in use or that it is damaged.

When you ask your computer to find a file, it uses the FAT to locate all clusters used by that file after the first.

Because damage to this table could be disastrous, the operating system keeps two copies, one immediately following the other. If the first goes bad, the second can be used to recover the data on the disk.

When you delete a file, all clusters associated with that file are re-marked as "not in use." This makes long files difficult to recover after accidental deletes.

The Root Folder

Immediately after the last sector used by the FAT, there appears a list of all files assigned to the root folder on the disk. This directory saves for each file the name, the date and time of creation, the attributes, the first sector and cluster used and the size. The number of sectors used for the root folder depends on the type of disk.

When you delete a file, only the first letter of the file name is changed. Because the rest of the information is still present, we may be able to recover the file after an erroneous delete operation. The missing data erased from the FAT, however, will make it difficult to recover long files.

The File Data Area

After the base folder comes the file data storage area, which fills the rest (almost all) of the disk. Double-sided drives for the *Atari ST*, for instance, use one sector for the boot record, five sectors for the FAT and 16 sectors for the base folder information. The remainder of the disk, a total of 1418 sectors, is available for data storage.

Summary of Disk File Storage

To use any file we need information from at least three different areas on the disk. The base folder stores the file name and attributes and the location of the first cluster of sectors used by the file. The FAT stores, for each cluster, the next cluster used to store the file, or a special character to show that no more clusters were used. Finally, the cluster or clusters within the file data area of the disk actually store the information.

A file is damaged if any of these areas is damaged.

The Keyboard

Several of the keys are assigned specific purposes in **MichTron Utilities**:

ARROW KEYS:	Used to move around in the viewed data
CONTROL/HOME:	Same as clicking on *Quit*
SHIFT/HELP:	Does an ASCII screen dump to the printer (FAST).
TAB:	Changes from ASCII/HEX modify.

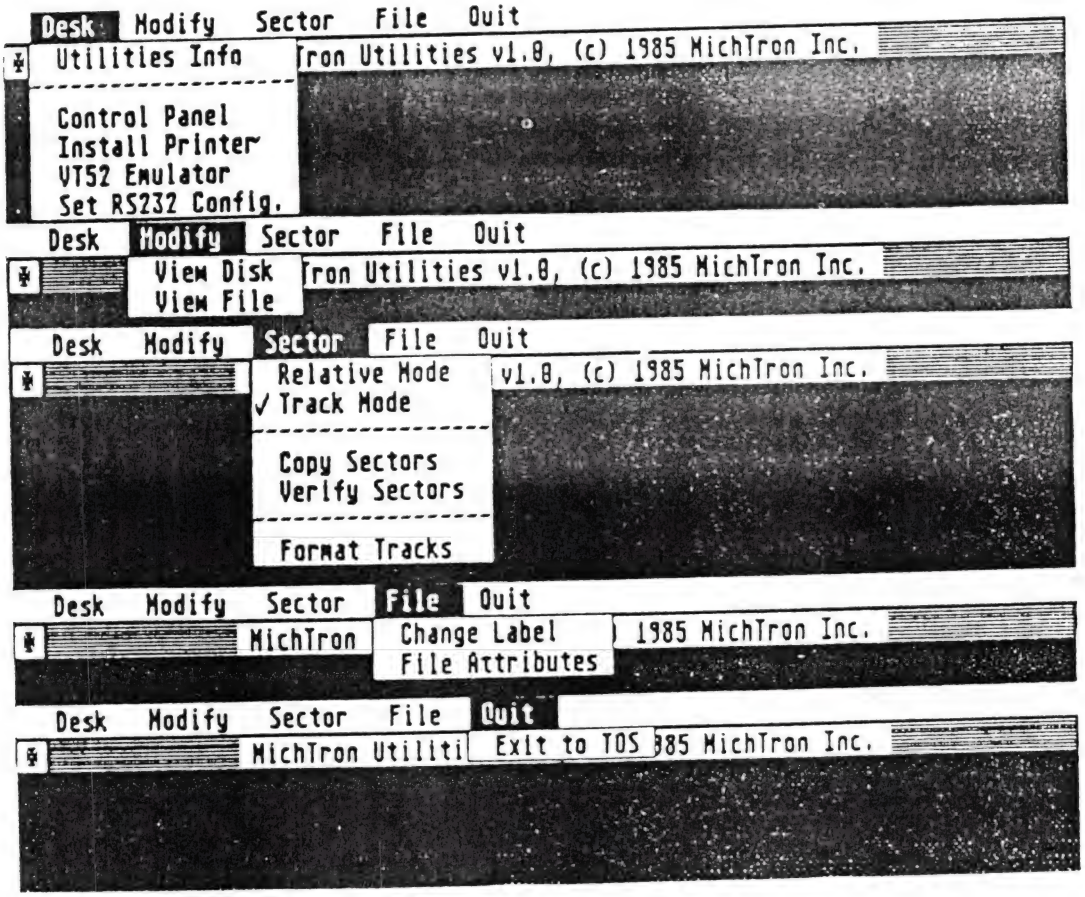
Chapter 2:

MichTron Utilities Functions

Functions

This section describes the individual functions available from **MichTron Utilities**. It does not usually provide application-specific information. If you want to repair a damaged disk, for instance, you will use several of these functions. Application procedures are outlined in the next major section of this part of the manual, **Common Operations**. Unless you are very experienced, you will need to read both sections before using this program.

All the utility functions are accessible from the opening screen. The figure below shows how the functions are grouped in the menus.



CHOOSE LOCATION Mode

You may choose to keep track of your location on a disk in either *Track* mode or *Relative* mode as you examine and change your disk contents. Each choice has advantages and disadvantages.

If you choose track mode, you must specify by number the track (and usually the sector of the track) you want to work with. You should certainly use this mode when you are planning to reformat one or more tracks. Since a whole track must be reformatted, rather than individual sectors, it's important for you to know just which sectors will be destroyed by the reformatting operation.

With track mode, you know just where you are looking on the physical structure of the disk.

If you choose *Relative* mode, you must specify the sector you want to work with by specifying the offset of that sector from the beginning of the file or disk. In many ways this is more convenient than actually keeping track of the physical layout of the disk files.

With *Relative* mode you know just where you are looking on the logical structure of the disk or file.

If you are addressing a floppy disk, you may choose the location mode you prefer for *VIEW DISK, COPY SECTORS* and *VERIFY SECTORS. FORMAT TRACKS* requires track mode and will automatically change to it if necessary. *VIEW FILE* requires *Relative* mode and will automatically change to it if necessary. Location mode is not relevant to other **MichTron Utilities** functions.

If you are addressing a hard disk, the program will automatically adopt the *Relative* mode. *FORMAT TRACKS* will not work on a hard disk.

VIEW DISK

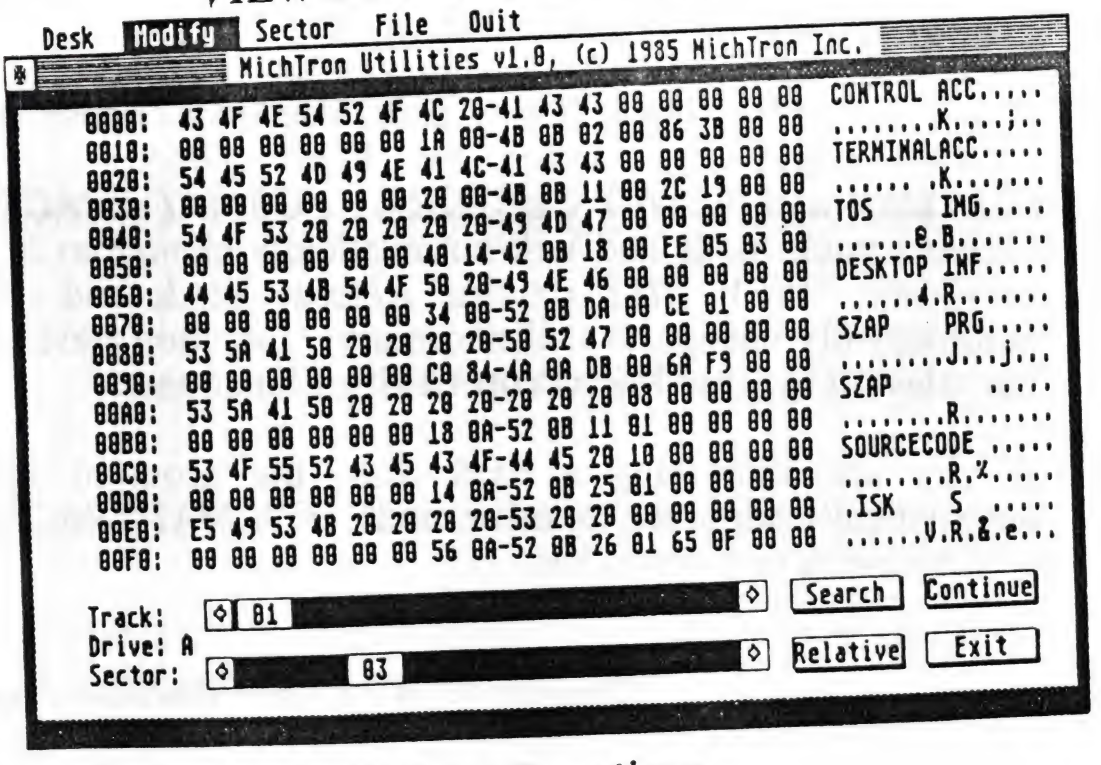
VIEW DISK is useful for examining and changing individual bytes in machine-language files and in the file control sectors at the beginning of each disk.

From the opening screen, select *VIEW DISK* by clicking once on the selector line. The screen will clear, then ask for a drivr specification. Part of the disk contents are displayed, and the controls for moving through and searching the disk.

Filling the top sixteen lines of the screen will be the contents of the first half of sector one, track zero of the default disk.

Below that you will find the *VIEW DISK* control area. At the lower left you will see the drive designation and one or two slider bars showing the current location on the disk. At the lower right you will find four function selector buttons.

VIEW DISK Screen in Track Mode

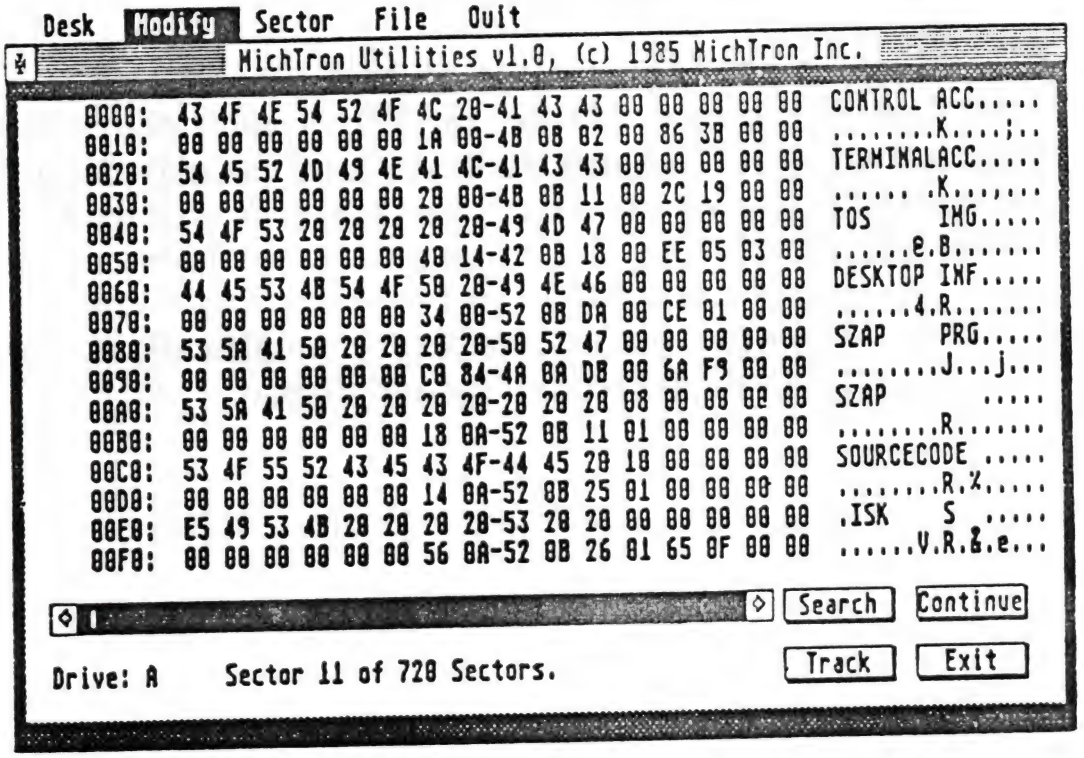


It will prompt for the drive to edit. Press ENTER to default to the last drive used. Select a Track or Sector by clicking on the respective object. Select a different drive by clicking on the word *Drive*.

Disk Contents Display

The screen shows the contents of half (256 bytes) of an individual disk sector. At the far left of each of the first sixteen lines is the relative address from the beginning of the current sector of the first byte shown on that line. In the middle of the line, the contents are displayed in hex code, on the right in ASCII, 16 bytes per line. You may read or modify either display; modifications are automatically duplicated in the second display.

***VIEW DISK* screen in *Relative* mode**



Viewing the Disk

If the disk you want to view is in the default disk drive, you don't need to do anything. If the disk is elsewhere, you may either transfer the disk to the default drive or change the view to another drive.

- To put a new disk in the default drive, make the disk change, click on the disk drive label in the lower left of the screen and press RETURN.

- To change the drive, move the pointer to the disk label in the lower left corner of the screen and click the mouse button once. Type the letter of the drive you want to view. The display will change to the new drive.

Choosing the Location Mode

On the lower right corner of the screen is a button labeled *Track* or *Relative* (opposite of your current location mode). To change location modes, just click your mouse on the button. It acts as a toggle switch.

Viewing the Track and Sector

Use the slider bars to move to the sector you want. The positions of the sliders represent the position of the current display within the disk. If you were in *Track* mode when you entered the *VIEW DISK* function, you will see slider bars for both track and sector; if you were in *Relative* mode, a single slider bar represents the relative location on the disk.

In *Track* mode, you may select a new track by clicking on and dragging the Track slider along its bar.

You may change the track number by 10 by clicking inside the Track slider bar area either before or after the Track slider.

- You may change the track number by one by clicking on the arrows at the ends of the Track slider bar.

- You may select a new sector by clicking on and dragging the Sector slider along its bar.

- You may change the sector number by 10 by clicking inside the Sector slider bar area either before or after the Sector slider.

- You may also change the sector number by clicking on the arrows at the ends of the Sector slider box. Each click there moves the display one-half sector forward or backward.

In *Relative* mode, you may select a new disk location by clicking on and dragging the Offset slider along its bar.

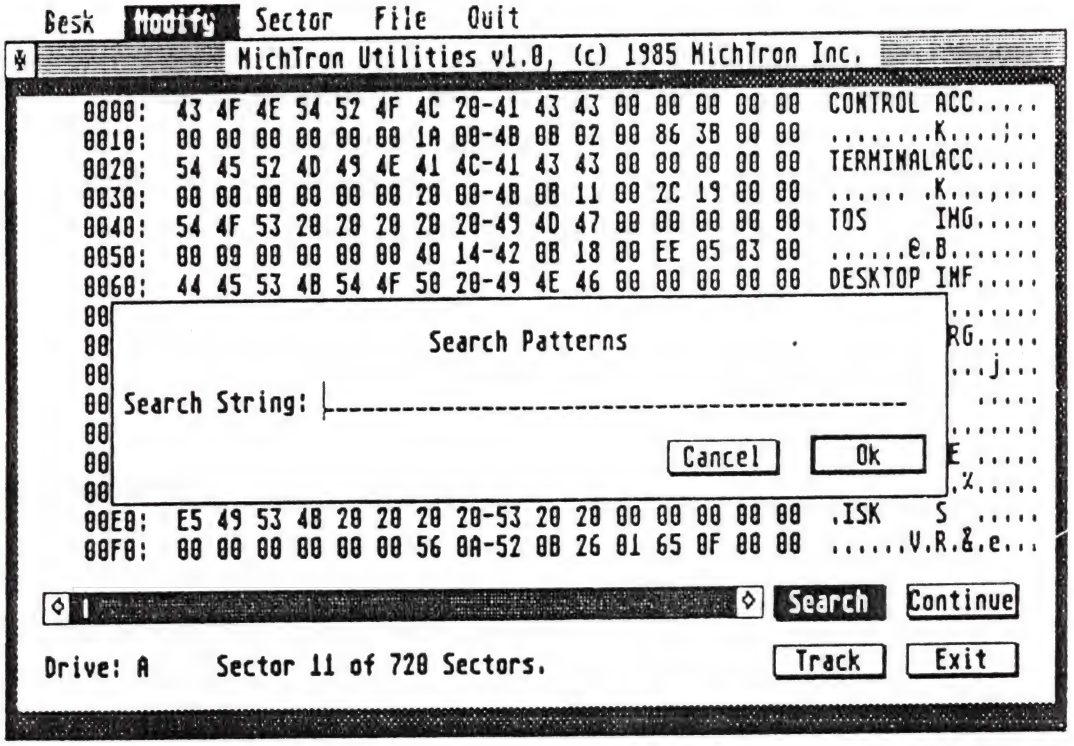
- You may change the sector number by 10 by clicking inside the Offset slider bar area either before or after the Offset slider.

- You may also change the current sector by clicking on the arrows at the ends of the slider box. Each click there moves the display one-half sector forward or backward.

When you move beyond the last or first sector of a particular track, the display moves to the next or previous track.

Searching for Character Strings

You may search for a string of up to 43 ASCII or 22 hex characters. Begin by clicking once on the *Search* button. A dialog box will appear with space to type your string.



Type in the string you want to search for and click on the *OK* button or press RETURN to start the search.

The default string type is hex. Each two keystrokes are counted as one hexadecimal number. For more readable entries, you may use commas to separate numbers. If commas are used, only a single digit is required. Even for zero, however, one digit is mandatory; a string of commas is interpreted as no entry.

The hex string mode is not case-sensitive.

If you want to enter an ASCII string, begin with either a single or a double quotation mark. All following text is interpreted as ASCII until the initial quotation mark is repeated. Both styles of quotation mark are allowed so that you may search for strings including one of them. The quotes need not be closed if the search string is entirely ASCII.

The ASCII string mode is case-sensitive.

Strings of mixed mode are allowed. Each ASCII section of the entry must begin with a quotation mark; if any hex entries follow, the ASCII section must be closed with the same quotation mark. You may use different quotation mark styles with different ASCII sections of a single search text.

In either mode, you may correct errors using BACKSPACE, DELETE, RIGHT ARROW and LEFT ARROW. ESC will erase the whole line.

If the program detects an illegal entry after you attempt to start the search, it will replace the first error with a dollar sign ($). You may correct the error and try again or cancel the search.

If the search succeeds, the screen will show the search text on a normal *VIEW DISK* screen with a cursor pointing to the text. If the entry was in hex, the cursor will be in the hex field; if the entry was in ASCII, the cursor will be in the ASCII field. If the entry was mixed in mode, the mode of the first letter determines where to place the cursor.

If the search fails, the program will display an alert. Click on the *OK* button. You will be left at the end of the disk or file.

Aborting a Search

If you decide to abort a search, press ESC, use the SHIFT keys, or hold the mouse button down. You will be left at the current position of the search.

Continuing a Search

If you want to repeat a search for the same string, just click on the *Continue* button on the lower right corner of the screen. The search will begin again immediately.

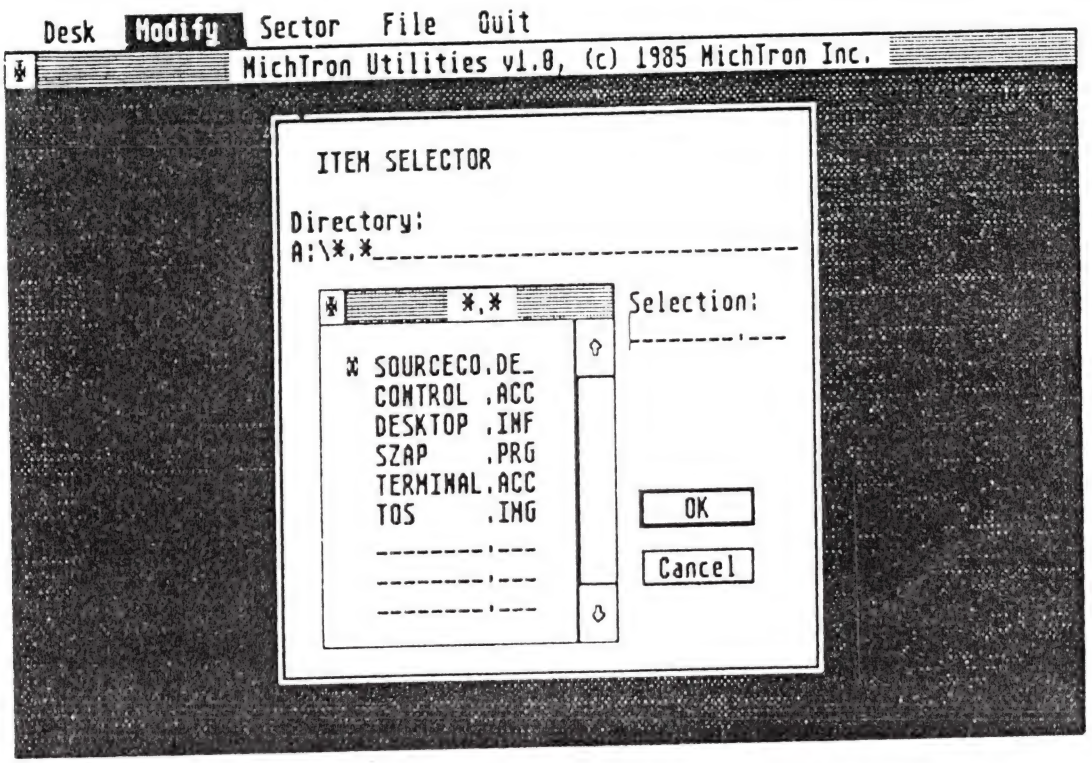
Quitting VIEW DISK

Click on the *Exit* button on the lower right of the screen. You will return to the opening screen.

VIEW FILE

VIEW FILE, like *VIEW DISK*, is useful for examining and changing individual bytes in files.

From the opening screen, select *VIEW FILE* by clicking once on the selector line. The screen will clear, then display the *Item Selector* dialog box. The screen names the current folder on the default drive near the top of the dialog box and displays its contents, or part of its contents, in the lower left. At the right you will see a place to type a file name. At the lower right are *OK* and *Cancel* buttons.



Viewing the Folder

To change the folder displayed, click on the line containing the name of the current folder. Backspace over the current information and type in the new folder name. Do not press RETURN. Instead, move the pointer to the folder identifier move bar just below the folder name and click once. The new folder loads immediately.

If the folder you want to view is part of the current folder, you may see its contents just by double-clicking on its selector line. You may open as many folders within folders as you like.

If you ask for a folder not present on the specified disk, the dialog box will display the root folder for that disk.

Viewing the File

Once you have the right folder, its files will be listed on the screen. If the number of files is large, use the vertical scroll bar to bring different files into view until you find the one you want. Choose the one you want to view by double-clicking on the file name or by clicking once and pressing or clicking on the *OK* button. You may also type in the file name if you want. *VIEW FILE* will remember the last path used to edit a file.

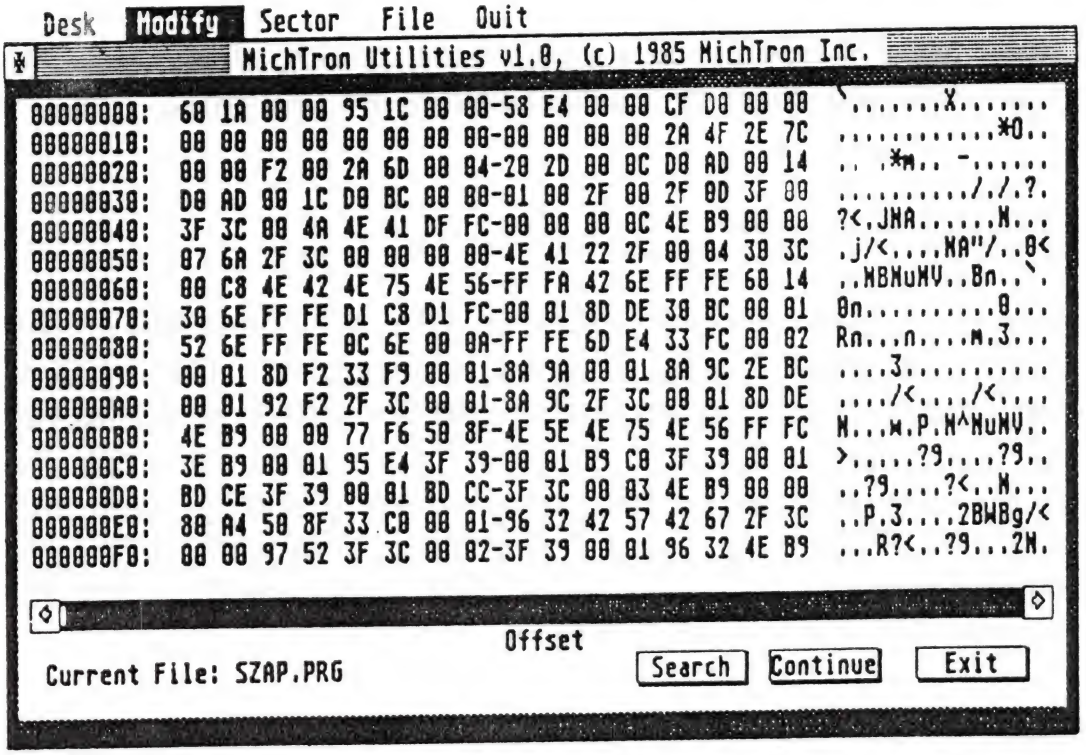
After you choose a file, the screen will clear, then display part of the file contents and the controls for moving through and searching the file.

Select a new file by clicking on the current file name.

File Contents Display

Filling the top sixteen lines of the screen will be the contents of the first 256 bytes of the file.

Below that you will find the *VIEW FILE* control area. At the lower left you will see the drive designation and file name. Above the name is an Offset slider bar showing the current location within the file. At the lower right you will find three function selector buttons.



The screen shows the contents of 256 bytes of the chosen file. At the far left of each of the first sixteen lines is the relative address from the beginning of the file of the first byte shown on that line. In the middle, the contents are displayed in hex code, on the right in ASCII, 16 bytes per line. You may read or modify either display; modifications are automatically duplicated in the second display.

Moving Through the File

You may look at other sections of the file by clicking on and dragging the Offset slider along its bar. The bar length represents the entire length of the file. The width of the slider shows (approximately) the fraction of the file displayed on the screen; its position shows the position of the display within the entire file. You can jump to the HEX offset by clicking on the word *Offset*.

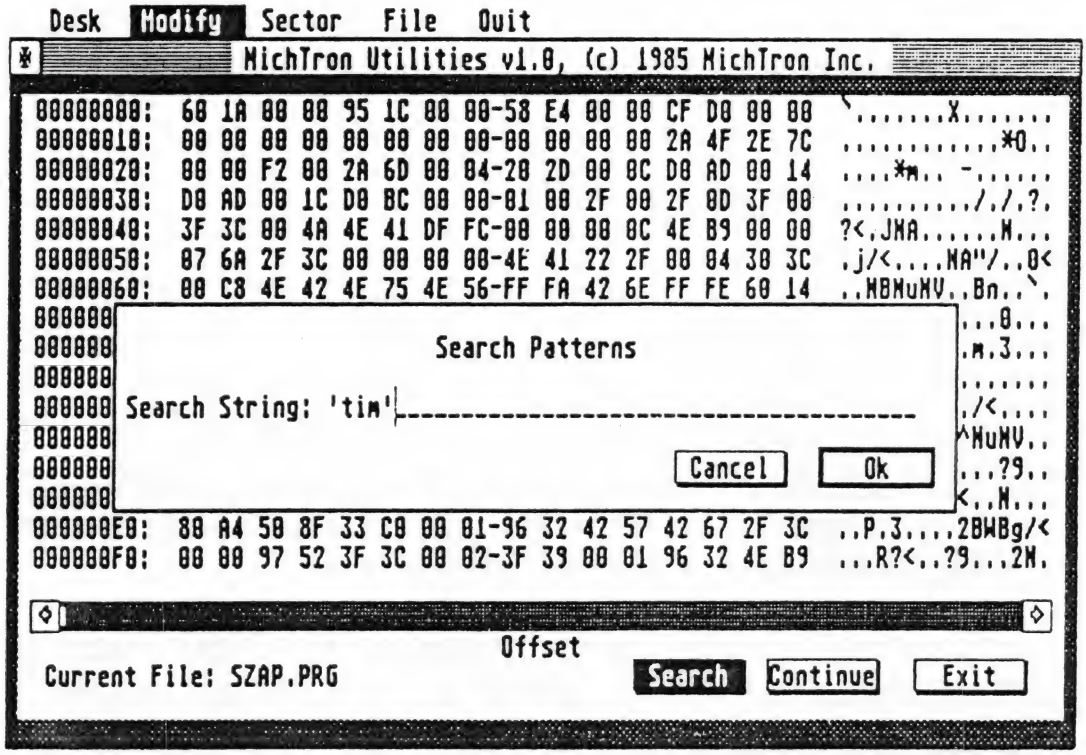
You may change the current display by 2560 bytes by clicking inside the Offset slider bar area either before or after the Offset slider.

You may also change the current display by clicking on the arrows at the ends of the slider box. Each click moves the display 256 bytes forward or backward.

Searching for Character Strings

> ***NOTE:*** *The* VIEW FILE/Search *function is identical in operation to the* VIEW DISK/Search *function, except that the area of search extends only over a chosen file, not the whole disk. The procedure is repeated here for convenience in reference.*

You may search for a string of up to 43 ASCII or 22 hex characters. Begin by clicking once on the *Search* button. A dialog box will appear with space to type your string.



Type in the string you want to search for and click on the *OK* button or press to start the search.

The default string type is hex. Each two keystrokes are counted as one hexadecimal number. For more readable entries, you may use commas to separate numbers. If commas are used, only a single digit is required. Even for zero, however, one digit is mandatory; a string of commas is interpreted as no entry.

The hex string mode is not case-sensitive.

If you want to enter an ASCII string, begin with either a single or a double quotation mark. All following text is interpreted as ASCII until the initial quotation mark is repeated. Both styles of quotation mark are allowed so that you may search for strings including one of them. The quotes need not be closed if the search string is entirely ASCII.

The ASCII string mode is case-sensitive.

Strings of mixed mode are allowed. Each ASCII section of the entry must begin with a quotation mark; if any hex entries follow, the ASCII section must be closed with the same quotation mark. You may use different quotation mark styles with different ASCII sections of a single search text.

In either mode, you may correct errors using BACKSPACE, DELETE, RIGHT ARROW and LEFT ARROW. ESC will erase the whole line.

If the program detects an illegal entry after you attempt to start the search, it will replace the first error with a dollar sign ($). You may correct the error and try again or cancel the search.

If the search succeeds, the screen will show the search text on a normal *VIEW FILE* screen with a cursor pointing to the text. If the entry was in hex, the cursor will be in the hex field; if the entry was in ASCII, the cursor will be in the ASCII field. If the entry was mixed in mode, the mode of the first letter determines where to place the cursor.

If the search fails, the program will display an alert. Click on the *OK* button. You will be left at the end of the file.

Aborting a Search

If you decide to abort a search, press ESC, use the SHIFT keys, or hold the mouse button down. You will be left at the current position of the search.

Continuing a Search

If you want to repeat a search for the same string, just click on the *Continue* button on the lower right corner of the screen. The search will begin again immediately.

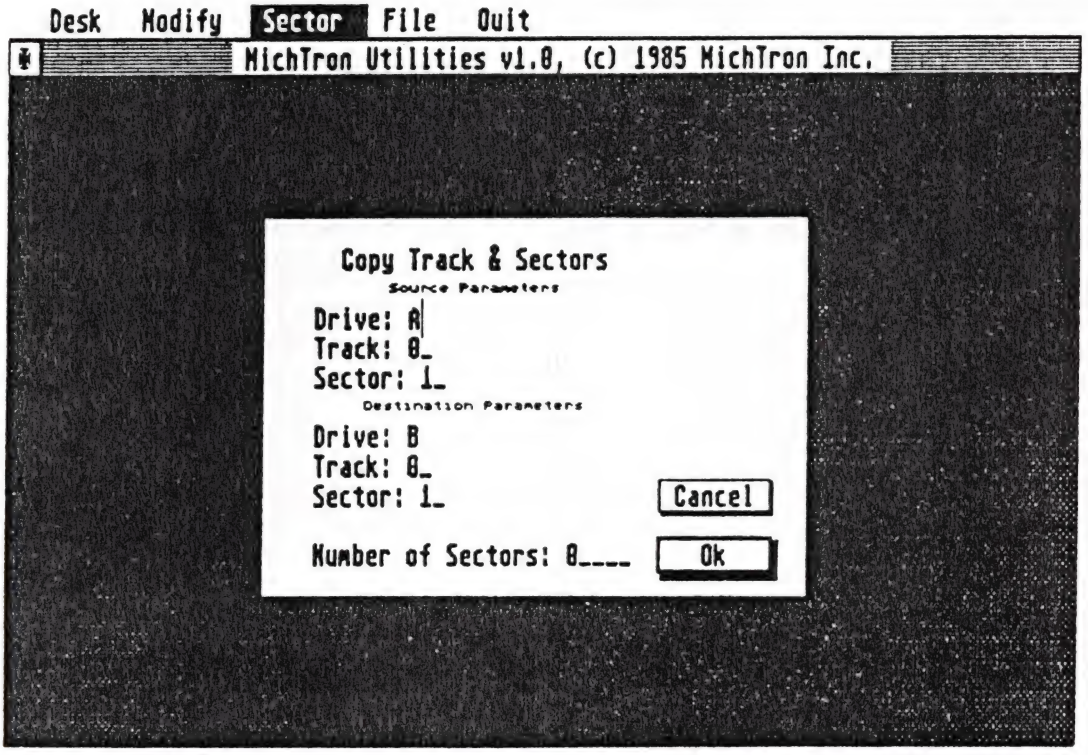
Quitting VIEW FILE

Click on the *Exit* button on the lower right of the screen. You will return to the opening screen.

COPY SECTORS

From the opening screen, select *COPY SECTORS* by clicking once on the selector line. The screen will clear, then display a dialog box in which you may specify the sectors to copy.

If you began this function while in track mode, you will see labels for drive, track and sector for both source and destination. Below will be a line to specify the number of consecutive sectors to be copied. At the right will be the *OK* and *Cancel* buttons.



If you began while in *Relative* mode, you will see labels for the drive and relative sector (from the first sector of track zero) for both source and destination. At the right will be the *OK* and *Cancel* buttons.

In either location mode, click on any label you want to change and type in the new information. When you have both source and destination properly specified, click on *OK* to begin the copy.

Copying begins immediately, so make sure you specify all parameters correctly.

A dialog box will appear to count down the sectors being copied. If copying succeeds, the dialog box will close up and the program will take you to the opening screen.

If a copy could not be made, an alert will warn you. Click on *OK* to return to the opening screen.

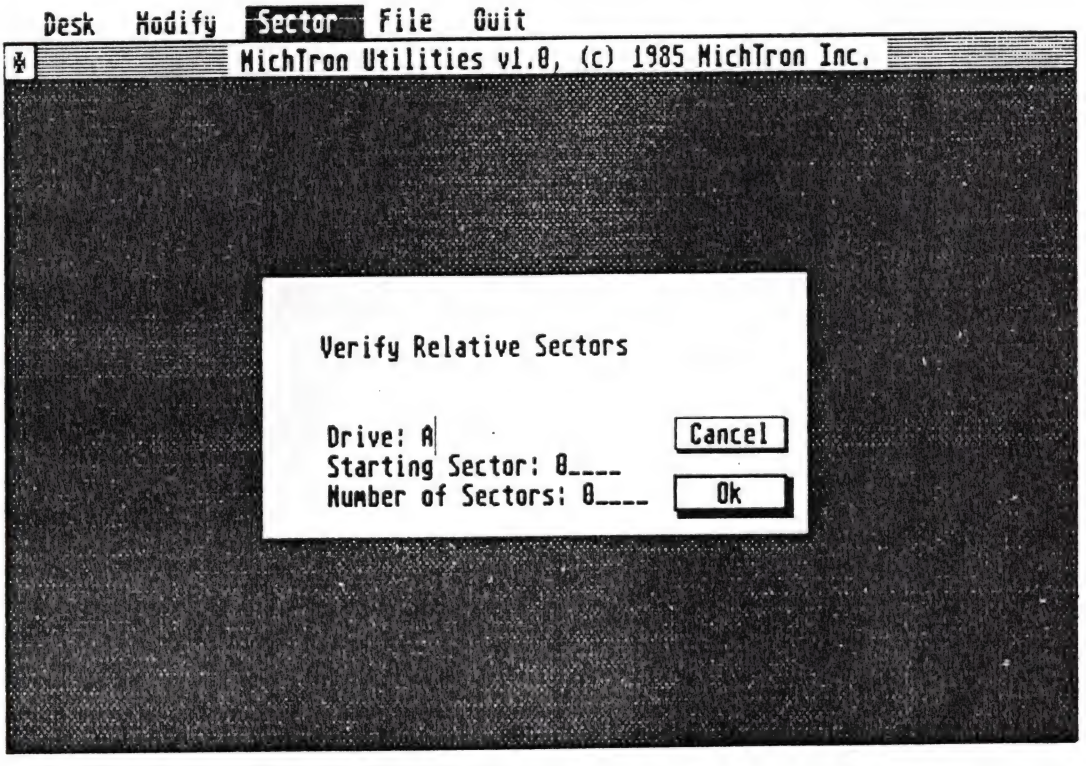
The function will abort if it encounters the end of the disk.

VERIFY SECTORS

From the opening screen, select *VERIFY SECTORS* by clicking once on the selector line. The screen will clear, then display a dialog box in which you may specify the sectors to verify.

If you began this function while in track mode, you will see labels for drive, track and first sector to be verified. Below will be a line to specify the number of consecutive sectors to be verified. At the right will be the *OK* and *Cancel* buttons.

If you began while in *Relative* mode, you will see labels for the drive and first sector (relative to the first sector of track zero) to be verified, as well as for the number of consecutive sectors to be verified. At the right will be the *OK* and *Cancel* buttons.



In either location mode, click on any label you want to change and type in the new information. When you have both source and destination properly specified, click on *OK* to begin the verification.

A dialog box will count down the sectors as they are verified. If the sectors checked are all usable, the dialog box will simply close automatically and the program will return you to the opening screen.

If any of the sectors checked is damaged, an alert will warn you. You may choose to continue verifying the other specified sectors by pressing *OK* or to abandon the verification by pressing *Cancel*. Choosing *Cancel* will send you back to the opening screen.

CLEAR SECTORS

This is a super-eraser that will overwrite any unassigned sectors with E5 characters (the characters written when a disk is formatted). Use this routine whenever you want to make absolutely certain that no one can recover data you have erased from a disk.

Any part of the disk that contains information from a deleted file will be permanently erased. You cannot recover any file deleted before the *Clear Sectors* Command.

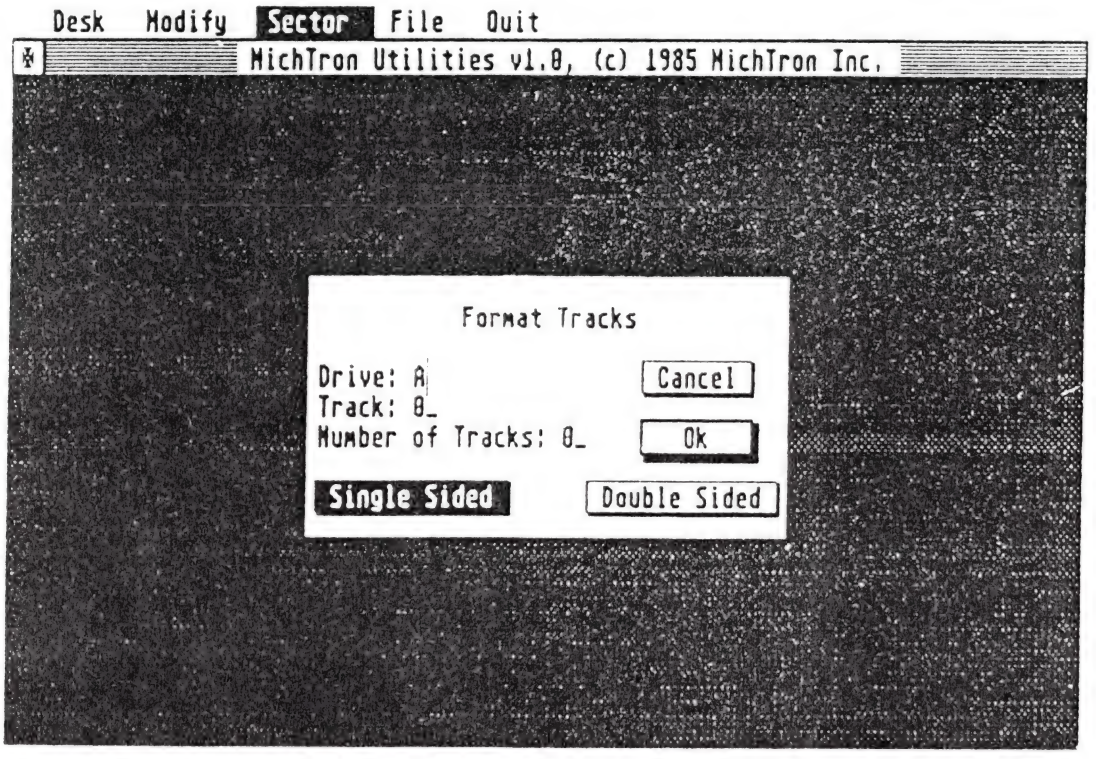
A dialog box will appear and you select a drive. A display will appear showing a map of the used/unused clusters. As the clusters are checked they are erased from the screen.

Press and hold either SHIFT key or hold down the mouse button to abort the function.

FORMAT TRACKS

From the opening screen, select *FORMAT TRACKS* by clicking once on the selector line. The screen will clear, then display a dialog box in which you can specify the tracks to format.

You will see labels for the drive and first track to be formatted. Below will be a line to select the number of consecutive tracks to be formatted. At the bottom you will see selector buttons for single- or double-sided disks. At the right will be *OK* and *Cancel* buttons.



Select between single- and double-sided by clicking on the appropriate button. To change the drive or track, click on the label you want to change and type in the new information. When you are sure you have selected the right tracks on the right drive (and that the right diskette is in that drive), click on the *OK* button.

A dialog box will count the tracks formatted. If all tracks are successfully formatted, the dialog box will close up and the program will return you to the opening screen.

If the formatting fails, an alert box will warn you. Click on *OK* to return to the opening screen.

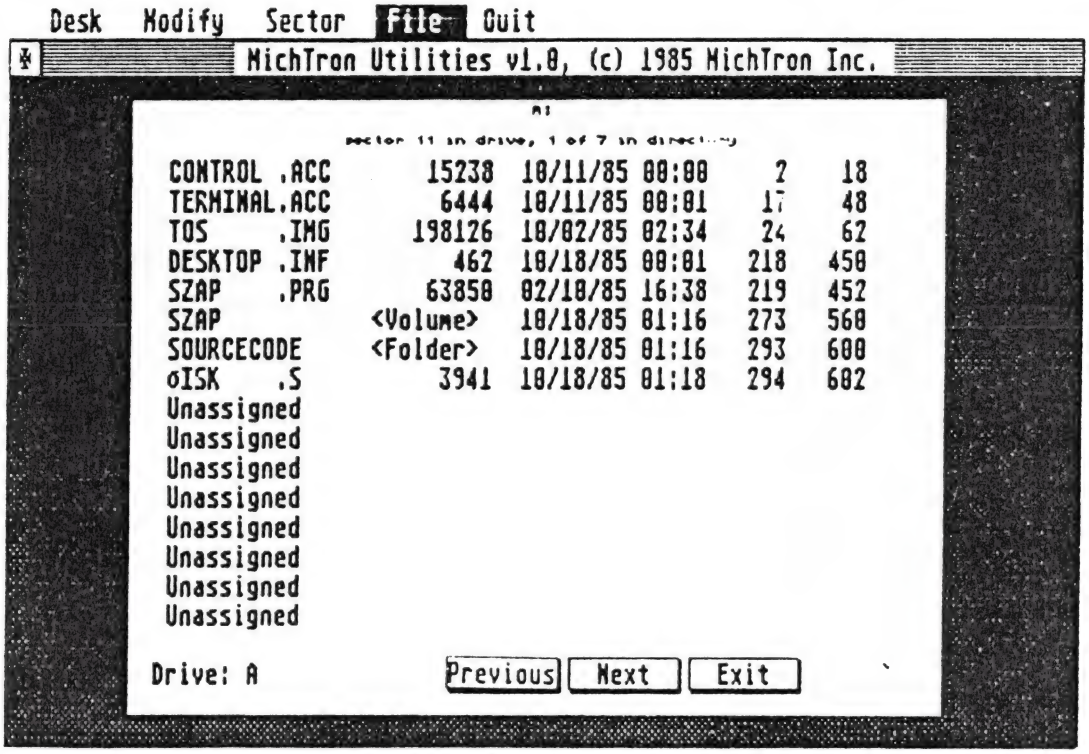
ABORT

To abort a *Search, Copy, Format*, or *Verify* function, hold down either SHIFT key, or hold down the mouse button while the function is executing.

FILE ATTRIBUTES

This function is used to alter the information about the file stored in the folder, but not the actual data stored in the file.

From the opening screen, select *FILE ATTRIBUTES* by clicking once on the selector line. The screen will clear, then display a dialog box listing the files belonging to the current folder.



Files are listed with their names and extensions, size in bytes, date and time of creation, starting cluster number and starting sector number. You may click on buttons at the bottom of the screen to see the next or previous group

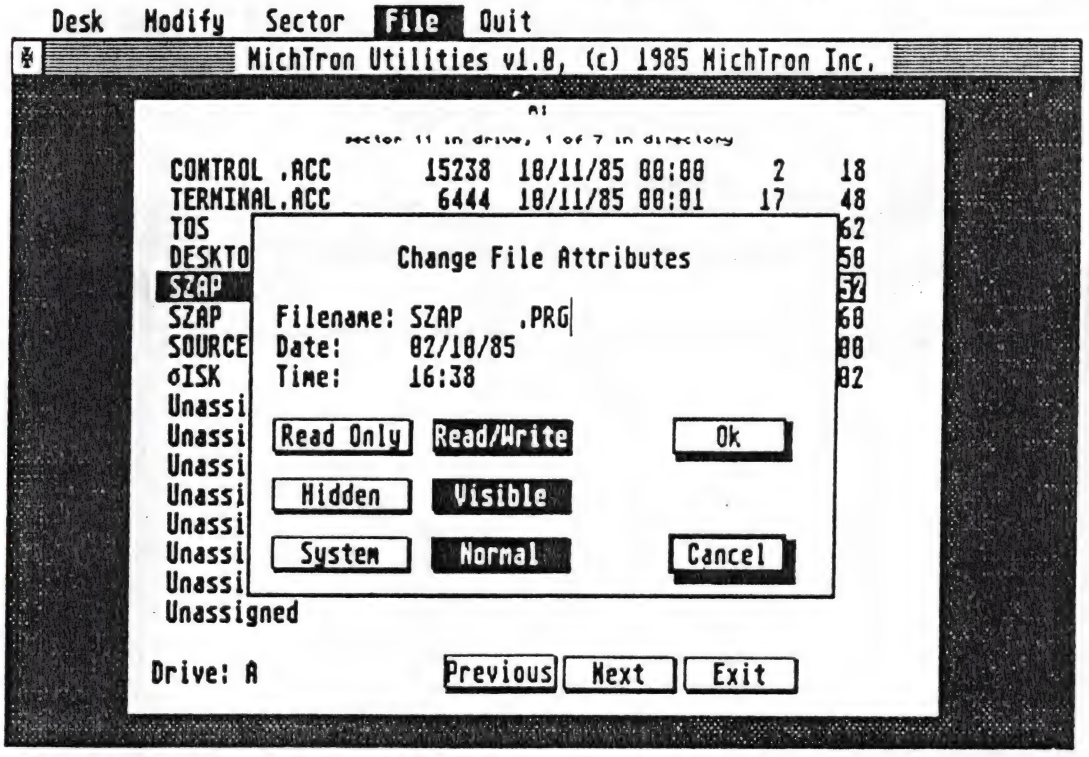
of files in the current folder. If you decide to abandon work, click on the *Exit* button.

The files listed may be active files, deleted files, folders or volume labels. Active files have complete file names and a non-zero size. Deleted files look similar, but the first character of their file name is replaced by a Greek delta. Folders and volume labels are labeled in the size column.

To select an item to work on, use the mouse to click on it. The program response depends on which item you picked.

Changing Active File Parameters

If you choose to work with an active file, click once on its line. A second dialog box will appear on top of the first.



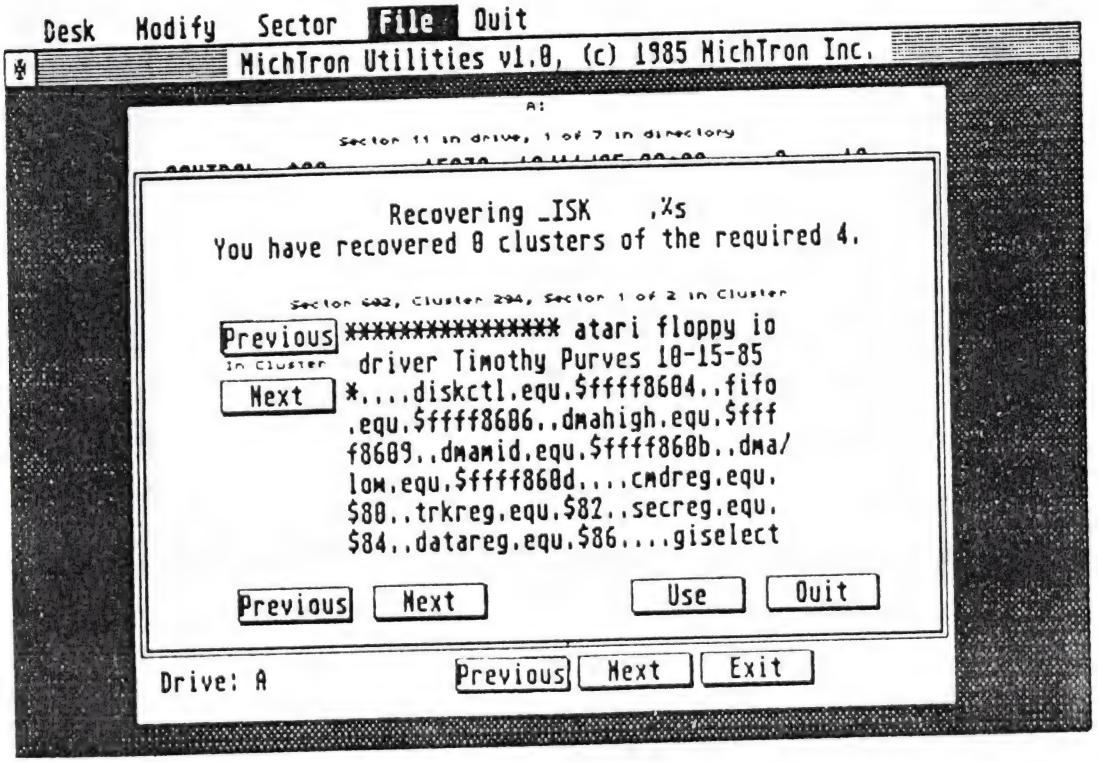
This box shows the file name and date and time of creation at the top. You may change any of these by clicking on the appropriate line, backspacing over the current information and typing in the revised data.

At the lower left you will also see pairs of buttons that set whether the file is read-only or read/write, hidden or visible, system or normal. You may change any of these by clicking on the appropriate button.

No change takes place immediately. You must click on the *OK* button on the right to actually make the changes you have entered. You may also, of course, *Cancel*.

Recovering Deleted Files

If you choose to try to recover a deleted file, click on its line to start. A second dialog box will show the file name and the number of recovered clusters at the top, then the current cluster and sector number and an ASCII display of the first sector of the file. Below and to the left will be a collection of buttons to control the search.



The upper pair of buttons will let you see the next or previous sector within the current cluster if such sectors exist. Looking at all the sectors may help you decide whether the material belongs to your deleted file.

The second row of buttons will let you choose to see the previous unassigned cluster or the next one if such clusters exist, to use the current material as part of the reconstructed file or to quit the attempt to reconstruct the deleted file.

The program searches for your deleted file by showing you every unassigned cluster on the disk, one at a time. You may make an immediate decision to include or reject a cluster or suspend decision until you have examined other sectors or clusters.

When all possible clusters have been inspected, the program will put up an alert to let you know you've seen everything. (You will not be shown unassigned clusters if the first sixteen characters of their first sectors are all format characters, because such characters would have been overwritten if the disk area had been used.)

The search ends only when you choose to quit. Watch the counter at the top of the screen to see when you have chosen enough clusters to reproduce the original length of the deleted file.

When you choose quit, an alert box will ask whether to save the clusters you have designated and quit, resume the recovery attempt or simply cancel without saving anything.

After you quit, with or without saving, the program will return you to the root folder of the current disk.

Opening a Folder

If you click on a folder, the folder will open to show its contents. If other folders are included, these can be opened as well. Clicking on either the .. or . entries in any folder will cause the program to return you to that folder's parent folder (not necessarily the root folder).

When it prompts for the drive to edit, pressing ENTER will default to the last used drive. Clicking on a folder name opens that folder. Clicking on a volume name allows you to change the name/date/time of the label name.

Clicking on a file beginning with a ? will attempt a recovery operation on that file.

Clicking on any other file entry allows you to change the name/date/time/attributes of the file. You may also create duplicate file names with this function. TOS, when it opens the file, will find the first one and not see the second one. This may be used to create special disks (limited only by your imagination).

DISK USAGE

This displays a graphic map of the usage of disk space on the specified drive. Black squares represent used space and white squares represent unused space. If there are a lot of white squares interspersed in the black the disk is fragmented.

It will prompt for the drive to edit. Press ENTER to default to the last used drive.

Clicking on the word *Track* or *Sector* on the *VIEW DISK* screen will produce a dialog box that lets you type in a numeric value for the specified item. If you know where you want to go, this may be faster than using the slider bar, especially on hard drives. Clicking on the word *Drive* pops up the *Default Drive* selector so you can change disks quickly.

> ***Note:*** *Because of quirks in the current* Atari *Operating System it is recommended that you re-boot the computer after recovering files, changing attributes or using* VIEW DISK *on the Hard Disk. This ensures that the Operating System memory tables are current and that all actions taken by* **MichTron Utilities** *are complete.*

Chapter 3:

Common Operations

Common Operations

This section outlines procedures for solving common problems and for making common alterations in your files. Generally, the procedures involve using one or more of the **MichTron Utilities** functions in the correct sequence and on the right targets. Details of the operations of the functions are given in the preceding section of this manual, **MichTron Utilities** functions. Unless you are very experienced in dealing with disk files, you should follow the procedures given here for solving problems.

Before you use any of the procedures for the first time, please try it out on information you are willing to lose. If you misread or misunderstand an instruction, it's important that you have a chance to find out you got it wrong before you destroy valuable data.

Always check to make sure that you issued your instructions with the right sources and targets specified and the right diskettes in the right drives.

Searching for Text

The most common reason to search for text is to find the files or portions of files containing a key phrase or string. This may be important if you cannot recall the name given to a file on a particular subject or if you are trying to recover lost sectors of information after a partial file loss.

To find a particular string of characters, whether ASCII or hex, select *VIEW DISK* or *VIEW FILE* as appropriate from the opening screen. Use the search sub-function described in the *VIEW DISK* or *VIEW FILE* functions. Their descriptions are duplicate and are provided in both places for easy reference.

Changing Text

The most common reason to change individual bytes on a disk is probably to patch machine-language programs for improved function or altered default values. Please do this only to a copy, not the original: an error in typing the replacement bytes could cause a fatal program error.

To change one or more bytes in a selected disk or file, use *VIEW DISK* or *VIEW FILE* to browse through the data or use the search sub-function to find the bytes to change.

To change any byte, simply move the mouse pointer to the beginning of the byte, click once to get a cursor on screen there and type in the replacement characters. If you are replacing more than one byte, keep typing; succeeding bytes will be replaced automatically as the cursor reaches them.

You may type in either the hex field of the display or the ASCII field. The other field will automatically be updated. If you try to enter an illegal character, the program will simply ignore it and wait for the next legal character to appear.

Watch the screen as you type and check before you approve storing your changes. Disaster awaits the unwary.

Changes are not permanent until you try to view another region of memory or exit the *VIEW DISK* or *VIEW FILE* function. In either case, a dialog box will give you the chance to save or cancel the changes. When you choose to save, the changes are made immediately.

Changing Volume Labels

You may want to change a volume label because the files now on the disk no longer correspond to the label originally given to it. Changing the volume label would be easier than copying all the files to a new disk with a better label.

To do so, use the *FILE ATTRIBUTES* function.

Changing File Names

You may want to change a file name to make the name more closely reflect file contents, to prevent confusion between similarly-named programs or to conceal the nature of a file from others.

To do so, use the *FILE ATTRIBUTES* function.

Changing FILE ATTRIBUTES

The file attributes, as outlined in the earlier section of this manual: *How Files Are Stored On Disk*, include flags to make the file a read-only file, a hidden file or a system file. Other file attributes are the date and time of creation or update. You may want to give a file one of these attributes, remove a file attribute, or change the creation time and date.

To do so, use the *FILE ATTRIBUTES* function.

Restoring Deleted Files

If you discover you have accidentally deleted a file you still want, drop whatever else you had in mind and immediately try to recover the file. If the first cluster of the file has not yet been overwritten, recovering a short deleted file is relatively simple. If the first cluster is already overwritten, recovering the rest of the file is still possible, but the process is not pleasant or quick.

To begin, select the volume on which the deleted file resided. Check to see if your file is still undamaged by using the *FILE ATTRIBUTES* function to look at the folder in which the file was stored. Deleted files appear on the file list with their first letter replaced by a Greek delta.

If the (truncated) file name does not appear on the list, recovering the file will be next to impossible. The first cluster will certainly have been overwritten and others may have been as well.

If the file name still appears, there is hope that you can recover the file. Click on the file name. The program will attempt to recover the first cluster of the file.

If the first cluster belongs to your deleted file, continue searching for the rest using the *FILE ATTRIBUTES* function.

If the first cluster of a file has been overwritten, the program will allow you to try to recover the rest, but a warning message will remind you that you are changing the starting location of the file and therefore cannot recover all the data.

Recovering Data from Damaged Disks

The usual reason for trying to recover data from damaged disks is to protect the remaining data before trying to salvage the original, damaged disk. This is important because reformatting even a single track is likely to wipe out several good sectors along with the bad ones.

Begin by using *VERIFY SECTORS* to find the bad sectors. Then use *COPY SECTORS* to preserve all the good sectors on another disk.

Repairing Damaged Disks

Begin by using *VERIFY SECTORS* to discover which sectors are damaged, which still readable.

Use *COPY SECTORS* to get the remaining information onto an undamaged disk.

Now you can use *FORMAT TRACKS* to reformat only those tracks on which the bad sectors appear.

Use *VERIFY SECTORS* again to see that the re-formatted tracks are now readable. If they are still bad, the disk cannot be repaired.

If the sectors are readable now, you can write back the information originally on them from a backup or, if you have no backup, at least restore the good sectors you saved before formatting the tracks.

Error Messages

Procedural errors and warnings will appear in alert boxes that give you the choice of continuing on despite the error or canceling the current operation. The following errors mean that a read or write operation failed.

Can't Read Drive X Disk Parameters

The file allocation table or root folder information on the disk in Drive X is damaged. The copy or verify attempt failed.

Error During Copy

A *Read* or *Write* error occurred while the *COPY SECTORS* function was active.

Error in Verify

A bad sector was found by the *VERIFY SECTORS* function.

Error Reading [Writing] X -

A read or write error occurred. Further information from the BIOS:

All Purpose Error
Bad Sector
CRC Error: There was a parity error on read or write.
Drive Not Ready: General Mishap
Media Change: A disk was removed and another put in its place without notifying TOS.
No Paper: This appears to be a TOS bug. Usually means bad sector.
Read Fault
Sector Not Found
Seek Error: The disk read/write head physically could not reach the location required.
Unknown Device
Unknown Error
Unknown Media: The computer does not recognize the disk format.
Write Fault
Write Protect: The disk addressed is write-protected.

Format Error

An error occurred while the FORMAT TRACKS function was active.

Sorry, I can't open X

The file X is hidden.

Warning: I am having problems reading the disk in drive X

The computer does not recognize the format of the disk, but it will assume the disk is a normal double-sided disk so that you may change the individual bytes in an attempt to fix things.

We have some real problems. Can't read SYSTEM DATA sectors. Error was in X sector in Y drive, A of B in directory

The file directory sectors or the FAT sectors or both are unreadable. The recover operation failed.

Chapter 4:

Supplemental Programs

Supplemental Programs

SNAPSHOT

On the *MichTron* disk you will find the program **Snapshot.**

Once you've installed this program, you can intercept any screen by pressing both the ALTERNATE and HELP keys at the same time. The program will write the current screen out to a file on Drive A: in the form *Snapxxxx.pix.*

The program will restart the file numbering at *Snap0000.pix* every time it's reloaded.

FORMAT

On the *MichTron* disk you will find the program **FORMAT.** This program lets you format disks for either the normal 9 sectors or 10 sectors per track. Using 10 sectors per track increases the usable disk storage from 360K to 389K on single-sided disks, and from 720K to 789K on double-sided disks.

Start the program by double-clicking on the program name or icon as usual. The screen will show a set of selector buttons and a slider bar to show you how much of the format is complete.

Choose which drive you want to use by clicking on the appropriate button. Then choose whether you want the Single- or Double-Sided format. Finally, choose the Normal or the Extended format by clicking on the appropriate button.

Click on *Format* to begin or on *Exit* to quit.

An alert box will tell you when the format is complete. Click on *OK* to return to the opening screen. If you want to format another disk exactly like the last one, simply click on Format.

> *Note: You cannot copy disks with Extended format by dragging the disk icons across the screen as usual. All read and write routines function normally - it's just the disk copy routine that fails.*

To make copies of disks with Extended format, you can use the **Mi-Dupe** normal backup. **Mi-Dupe** automatically adjusts for either 9- or 10-sector formats. You may also, of course, copy the individual programs normally.

M-COPY

This lets you duplicate *Atari* disks in the least amount of time. Any program up to about 150 kilobytes can be duplicated on a single-sided disk.

M-Copy saves time by formatting and copying only those sections of the disk that are actually used by the program. The rest of the disk is left unformatted. Time is saved on every duplication, with short programs saving the most time.

Mi-DUPE

This is a program that allows you to backup almost all your disks, even those with copy protected material on them.

Whether you want a quick copy routine to duplicate your data files or a special program to archive even your copy-protected software, **Mi-Dupe** is for you.

Fully menu-driven, **Mi-Dupe** is a snap to learn and easy to operate even if it's been months since you last used it.

Mi-Dupe runs on any *Atari ST* with at least one disk drive.

Loading Mi-Dupe

1. Boot up with TOS from disk or from ROM.
2. Insert your working copy of **Mi-Dupe** into one of your disk drives. If necessary, press ESC on your keyboard to alert the computer to the change in disks.
3. Double-click on the MIDUPE.PRG text line or icon.
4. The **Mi-Dupe** opening screen will appear.

Making a Normal Backup

To copy your data files or other unprotected disks, choose the *Normal Backup* option from the *OPTIONS* menu.

Because this copy routine duplicates only those sectors actually in use on the source disk, it is often much faster than the TOS Disk Copy routine.

Normal Backup is also handy because it will do its own formatting if necessary. You can take a new disk right out of the package and copy data onto it.

Choose *Normal Backup* by clicking once on its selector line in the *OPTIONS* menu. A dialogue box will appear. At the top are selector buttons to assign the disk drives for the source and destination disks. Below are buttons to choose whether to format the disk before copying. The progress bars will show you how much longer the copy will take. At the bottom are the *OK* and *Cancel* buttons.

You may choose any combination of drives by clicking on the selector buttons. If you use the same drive for source and destination, alert boxes will ask you to switch disks as necessary. After you swap disks, click on *OK* to continue.

If you choose Y for formatting, the program will format the destination disk before it copies. If you choose N, the program will try to copy the disk sector by sector. If it finds a bad sector on the destination disk, an alert box will tell you the copy attempt aborted. Click on *OK* to begin again.

When you have correctly chosen the source and destination drives and chosen whether to format or not, click on *OK* to begin. The progress bars will fill with color as the copy proceeds. The upper bar traces reading from the source disk, the lower bar writing on the destination disk.

If you decide not to make a copy, click once on *Cancel.*

Making a Special Backup

To back up material on copy-protected disks, choose the *Special Backup* option by clicking once on its selector line in the *OPTIONS* menu.

Because this copy routine must inspect the disk to determine its non-standard format, normal Backup is faster on any unprotected disk.

The destination disk will automatically be formatted to match the copy-protection scheme.

A dialogue box will appear. At the top are selector buttons to assign the disk drives for the source and destination disks. Below these are the progress bars and the *OK* and *Cancel* buttons.

You may choose any combination of drives by clicking on the selector buttons. If you use the same drive for source and destination, alert boxes will ask you to switch disks as necessary. After you swap disks, click on *OK* to continue.

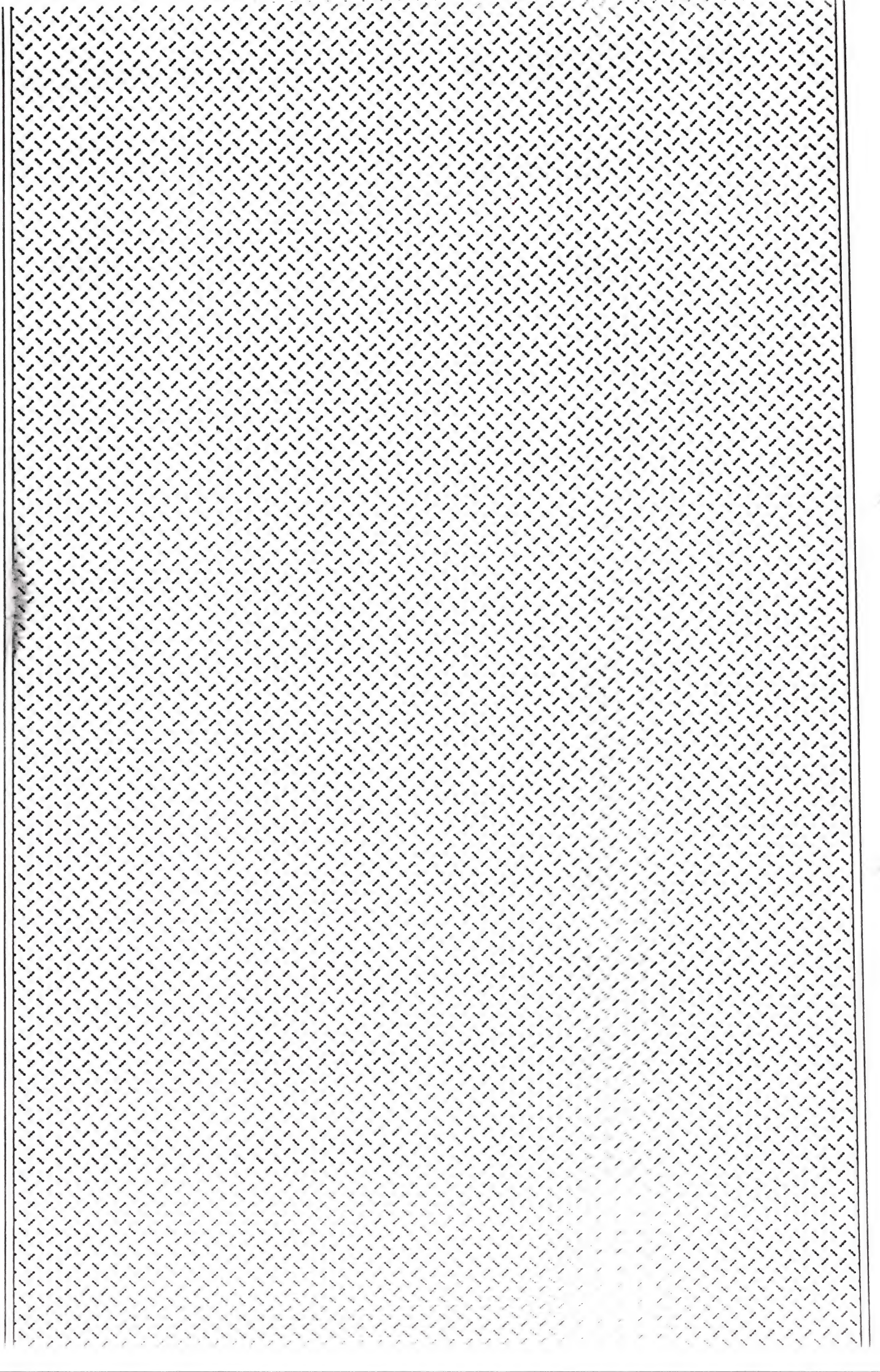
When you have correctly chosen the source and destination drives, click on *OK* to begin. The progress bars will fill with color as the copy proceeds. The upper bar traces reading from the source disk, the lower bar writing on the destination disk.

If you decide not to make a copy, click once on *Cancel.*

We cannot guarantee that *Special Backup* will produce a usable duplicate of a disk. Some protection schemes may be introduced that have such exotic formats that *Special Backup* cannot properly duplicate them. Certain physical modifications to disks will also keep this program from making a working duplicate.

Quitting the Program

To exit from the program back to TOS, just click once on the *Quit* selector line in the *OPTIONS* menu. Since you cannot select *Quit* while a copy operation is in progress, transfer to TOS is immediate.



Part II

A Collection of

21 Indispensable Programs

and Desk Accessories

for the Atari ST

Written by J. Weaver, Jr. and Timothy Purves

Chapter 5:

An Introduction to STuff

Introduction

STuff is a collection of programs and desk accessories for your *Atari ST* computer. These programs have been carefully designed to minimize the hassle and maximize the benefits of your ST. STuff contains programs for all levels of ST user, from the novice to the advanced professional, and for all ST systems, from the smallest to the largest. All of the programs have been tested together and are completely compatible with each other, as well as with most other non-copy-protected commercial software.

> ***NOTE:*** *Different programs in* **STuff have different hardware requirements. Some may require a color or monochrome monitor, while some may require the use or absence of a hard drive, or some other peripheral. Please check the requirements for each individual program (listed at the beginning of the section that contains the instructions for that program) before installing or using it.**

The **STuff** manual is divided into five sections for each of the five different types of programs included:

AUTO STuff: For programs designed to be used in your *ST*'s AUTO folder

DESK STuff: For GEM desk accessories

GEM STuff: For GEM application programs

TOS STuff: For TOS programs (which do not use the GEM interface)

TTP STuff: For TOS programs that require parameters

The first part of each section gives a short description of each program in that section, as well as generic installation and instructions for use of these programs. Following these generic instructions are specific instructions for each individual program included in the section, in alphabetical order.

NeoChrome Compatibility

NEOCHROME users will notice the color jiggles on the screen during mouse movement whenever **CAPSLOCK**, **KEYCOMBO** and/or **ONEHAND** are installed. Other than the visual effect, this will not affect the operation of the program in any way.

To our knowledge, **NEOCHROME** is the only popular ST program affected in this way, which is caused by the interaction between NEOCHROME's multi-color screen routines and the keyboard intercept drivers of the three STuff programs.

If the effect becomes unpleasant, we suggest that you use the ***ST*SELECT** utility in the **STuff** collection to disable **CAPSLOCK, KEYCOMBO** and **ONEHAND** during your **NEOCHROME** sessions.

Chapter 6:

AUTO STuff

AUTO STuff

This section contains programs designed to be used in your *ST*'s AUTO folder. AUTO folder programs are automatic — they're executed every time your computer is powered up or reset, before the Desktop is started, in the order in which they were placed into the AUTO folder. Some common AUTO folder programs include RAM disks, printer spoolers and clock-card drivers.

In this section, you'll find the following AUTO folder programs:

AUTODATE: For those of you without battery-backed clock cards, this is a program to ease the chore of manually typing in the date and time. **AUTODATE** remembers the last date your computer was booted up, and allows you to easily modify the date and time, using the arrow keys.

CAPSLOCK: This is a program for people who are always bumping the CAPSLOCK key by mistake. With **CAPSLOCK** installed, the CAPSLOCK key is disabled unless you are also hold down the ALTERNATE key.

HARDAUTO: This is for hard drive users. It's not an autobooter, but it's close: HARDAUTO allows your system's AUTO folder to be located on your hard drive, for faster access — a real time-saver, especially if you reboot often.

HIGH: This program, for color monitor users only, simply changes your screen resolution from its default (Low Resolution) to Medium Resolution mode. For other AUTO folder programs which may require (or look better in) the Medium Resolution mode, **HIGH** is a life-saver.

KEYCOMBO: This program sets up three different "hot keys" on your *ST*. When **KEYCOMBO** is installed in your AUTO folder, pressing ALTERNATE/UNDO sends the standard ASCII form feed character to your printer, ALTERNATE/DELETE blanks your monitor's screen (without affecting any underlying programs), and either CONTROL/ALTERNATE/ DELETE or CONTROL/ALTERNATE/ SHIFT/ DELETE reboots the machine.

ONEHAND: This program turns the ALTERNATE, CONTROL and SHIFT keys into "on-off" toggles, just like the CAPSLOCK key. While primarily designed for use by the handicapped, ONEHAND also has other uses — for example, using it to lock the Alternate key "on" allows you to move the mouse pointer with the keyboard with only one hand.

RESET: This is a memory cleaning program. With **RESET** in your AUTO folder, every reset of your computer (pressing the reset button on the back of the computer) will actually be a "cold boot" (just like turning the computer off and then back on). This allows memory to be resized, and can also be used to remove or reinitialize "reset-proof" resident

programs (like some RAM disk programs) without the possible thermal stress to the ST of cycling the power switch.

***ST*SELECT:** Now that you have so many AUTO folder programs, you may have a problem — you may not always want all of them operative at the same time. ***ST*SELECT** is designed to let you easily choose which of your AUTO folder programs and desk accessories to use every time you reboot your computer.

VERIFY: This AUTO folder program speeds up most *ST* applications by disabling the system's automatic verification of disk writes.

All of these programs will work with any *ST* system, except **AUTODATE** (which requires that you do *not* have a clock card), **HARDAUTO** (which requires a hard drive), and **HIGH** (which requires a color monitor).

Installing AUTO STuff

To install AUTO folder programs on your *ST* system, copy each program file you want to install (all of which have the extension PRG) from the **STuff** distribution disk to the AUTO folder of the disk you use in Drive A to boot up your system. If you're using a hard drive with an autoboot program (such as the June 16, 1986 version of AHDI) which does not require a disk in Drive A, your AUTO folder should be on your hard drive. If you're using **HARDAUTO** and a hard drive, you should have AUTO folders on both the floppy in Drive A *and* your hard drive — see the **HARDAUTO** installation note, below.

If your disk does not already have an AUTO folder, create one using the *New Folder...* command from the Desktop. Open a window to the root directory (that is, not inside any folders) of the proper disk, and then select *New Folder...*, type AUTO at the prompt, and click *OK*, or press RETURN.

The order in which files are copied to the AUTO folder (technically, the order in which they appear in the disk's directory) is the order in which they are executed. Certain **AUTO STuff** programs work better in specific locations in the AUTO folder.

If you use a hard drive, any driver program for the hard drive you require (AHDI, SUPBOOT, or the like) should be the first program in your AUTO folder. If you're using **HARDAUTO** with your hard drive, that should be next, to save the maximum amount of time possible. When using **HARDAUTO**, the hard disk driver and **HARDAUTO** itself should be the only two programs in the AUTO folder on Drive A — all the rest of your AUTO folder programs

should be installed in the AUTO folder on the hard drive itself.

***ST*SELECT** should be placed next, to maximize its effectiveness. It can be placed anywhere in the AUTO folder, but should be before any other AUTO folder programs that you may wish to select or deselect by using it.

AUTODATE can be placed anywhere in the AUTO folder, but should be before any other AUTO folder programs which require the system to know the correct date and time.

CAPSLOCK, **KEYCOMBO**, and **ONEHAND** can be placed anywhere in the AUTO folder, but should be before any other AUTO folder programs which modify the system TRAP #13 and TRAP #14 handlers (none of the **AUTO STuff** programs, including these three, actually alter these vectors, but other AUTO folder programs do, including MICHTRON's **Soft-Spool** printer spooler).

HIGH can also be placed anywhere in the AUTO folder, but should be before any other AUTO folder programs which require the system to be in medium resolution (none of the other **AUTO STuff** programs require this, but several of them will look substantially different — a bit of experimentation with the order of **HIGH** in the AUTO folder will yield the combination that looks the best to you).

VERIFY can be placed anywhere in the AUTO folder.

Finally, be sure to look at **AUTOFOLD** (in the **GEM STuff** section) for an easy way to change the order of AUTO folder programs.

AUTO STuff Programs

AUTODATE

When your *ST* system is reset or powered up, **AUTODATE** will automatically clear the screen, and display the date set the last time your computer was reset or powered up. The screen will look something like this:

```
Current date is: Thu, Mar 19, 1987
Enter new date: Thu, Mar 19, 1987
```

The lower line of the screen is where any changes made to the system date will be displayed.

AUTODATE uses an initial 5-second timeout — in other words, if no key is pressed within 5 seconds after the **AUTODATE** screen is displayed, the program assumes that no changes in the date or time are desired, and continues the boot-up process, setting the system date as shown, and the system time to 12:00 am (midnight). Pressing *any* key will abort the 5-second timeout.

To change the system date, use the following keys:

UP ARROW	This subtracts 1 month from the system date.
DOWN ARROW	This adds 1 month to the system date.
LEFT ARROW	This subtracts 1 week from the system date.
RIGHT ARROW	This adds 1 week to the system date.
BACKSPACE	This subtracts 1 day from the system date.

SPACE	This adds 1 day to the system date.
UNDO	This restores the previous date shown above the new system date (in case of grievous editing mistakes).
RETURN	This ends the "date" portion of **AUTODATE**, and starts the "time" portion.

When the "date" portion of the program is ended, you will see the following two lines on the screen:

```
Current time is: 12:43 pm
Enter new time: 12:43 pm
```

The upper line shows the time you set the last time the system was reset or powered up. The lower line is where any changes made to the system time will be displayed.

To change the system time, use these keys:

UP ARROW	This subtracts 1 hour from the system time.
DOWN ARROW	This adds 1 hour to the system time.
LEFT ARROW	This subtracts 10 minutes from the system time.
RIGHT ARROW	This adds 10 minutes to the system time.
BACKSPACE	This subtracts 1 minute from the system time.
SPACE	This adds 1 minute to the system time.
UNDO	This restores the previous time shown above the new system time (in case of grievous editing mistakes).

CLR HOME	This restarts **AUTODATE**, allowing you to re-edit the date.
HELP	This toggles between 12 and 24 hour display modes.
RETURN	This exits **AUTODATE**, saving the current date and time selected as the default for the next time the system is reset or powered up, and setting the system date and time.

Technical Information

AUTODATE saves the system date and time by using its own file date and time as the save area. In this way, the program itself does not need to be modified; however, because a change is being made to the program disk, that disk must *not* be write-protected when **AUTODATE** is run. In addition, you should *never* rename the AUTODATE.PRG file yourself (other than with STSELECT), or it will not be able to find itself in the AUTO folder, and will not operate correctly.

CAPSLOCK

When your *ST* system is reset or powered up, **CAPSLOCK** changes the way that your system responds to the CAPSLOCK key. Pressing the CAPSLOCK key by itself will do absolutely nothing, other than generate a key click sound. To toggle the **CAPSLOCK** function on or off, you must hold down the ALTERNATE key while you press the CAPSLOCK key.

To help you remember which state the **CAPSLOCK** function is in, it makes two different sounds through your monitor's speaker when the **CAPSLOCK** function is toggled on and off. The on sound, the higher pitched of the two, is very much like the standard *ST* key click, except that it is exactly one octave lower in pitch than the standard key click sound. The off sound, the lower pitched of the two, is also a key click, this one two octaves below the standard key click.

HARDAUTO

When **HARDAUTO** is executed from your AUTO folder, any programs in the AUTO folder of your hard drive are immediately executed, in the order in which they appear in the directory of that folder. When all PRG files in the hard drive's AUTO folder have been executed, the remainder of the programs (if any) in the AUTO folder on Drive A are executed.

HIGH

When **HIGH** is executed from your AUTO folder, your color *ST* system is immediately put into the medium resolution mode (640 by 200 pixels, 4 colors), instead of the default low resolution mode (320 by 200 pixels, 16 colors). The system will remain in that mode, unless specifically changed by another program (for example, the Desktop, by using the *Set Preferences...* dialog in the *OPTIONS* menu).

KEYCOMBO

When **KEYCOMBO** is installed in your AUTO folder, three different "hot key" combinations are enabled:

ALTERNATE/ UNDO	This sends the standard ASCII form feed character (12 decimal, or 0C hex) to your printer.
ALTERNATE/ DELETE	This blanks the current screen display, without otherwise affecting any program(s) in progress. Pressing any key when the screen is blanked restores the previous display.
CONTROL/ ALTERNATE/ DELETE	This performs a "warm boot" of the *ST* (just like pressing the *Reset* button on the back of the computer).
CONTROL/ ALTERNATE/ SHIFT/ DELETE	This performs a "cold boot" of the *ST* (just like turning the computer off, and then back on again). You must use the left SHIFT key.

ONEHAND

When your *ST* system is reset or powered up, **ONEHAND** changes the way that your system responds to the ALTERNATE, CONTROL and SHIFT keys. Each of these keys is now an on/off toggle, much like the CAPSLOCK key is. Pressing any of these keys once turns that function on; pressing that key again turns it off. Any number of the functions can be on at any one time. Please note that LEFT SHIFT and RIGHT SHIFT are treated separately — if you press LEFT SHIFT to toggle in the shifted characters, you must press LEFT SHIFT again to toggle them off. The same holds true for the RIGHT SHIFT key. This was done because some application programs distinguish between the two SHIFT keys, and have different functions for each of them.

To help you remember which state the special keyboard functions are in, **ONEHAND** makes two different sounds through your monitor's speaker when the functions are toggled on and off. The on sound, the higher pitched of the two, is very much like the standard *ST* key click, except that it is exactly one octave lower in pitch than the standard key click sound. The off sound, the lower pitched of the two, is also a key click, this one two octaves below the standard key click.

RESET

When you use **RESET** in your AUTO folder, each warm boot of your computer (pressing the RESET button on the back of the machine) will actually result in a "cold boot", which will entirely reinitialize the computer. Any "reset-proof" resident programs (such as certain RAM disk programs) will be removed.

RESET can also be executed as a TOS program from the Desktop to restore full memory usage after the 512K program has been used.

*ST*SELECT

When your *ST* system is reset or powered up, ***ST*SELECT** will automatically clear the screen, and display a list of all AUTO folder programs and desk accessories available on your system. The list will be divided into two columns: the left column displays the AUTO folder programs, while the right column displays the desk accessories. Programs or accessories which are currently selected to be used are displayed in white letters in a black box, while those programs or accessories currently not selected are displayed in black letters. The letters in parenthesis in the column headers show which drives the AUTO program and accessory listings were read from.

*ST*SELECT uses an initial 5-second timeout — if no key is pressed within 5 seconds after the ***ST*SELECT** screen is displayed, the program assumes that no changes in the AUTO folder programs and accessories selected is desired, and continues the boot-up process. Pressing *any* key will abort the 5-second timeout.

To select and de-select AUTO folder programs and accessories, ***ST*SELECT** uses the following keys:

UP ARROW and DOWN ARROW	These keys move the cursor (between the columns) up and down.
LEFT ARROW	This key changes the state (selected or de-selected) of the AUTO folder program immediately to the left of the cursor.
RIGHT ARROW	This key changes the state (selected or de-selected) of the desk accessory immediately to the right of the cursor.
RETURN	Pressing this key exits ***ST*SELECT**, saving the current state of all AUTO folder programs and desk accessories.

Technical Information

***ST*SELECT** selects and de-selects AUTO folder programs and desk accessories by changing the last character of the extension of the program or accessory file. A de-selected AUTO folder program is given the extension PRX, and a de-selected desk accessory is given the extension ACX. Since the TOS only recognizes AUTO folder programs with the extension .PRG, and desk accessories with the extension .ACC, this allows ***ST*SELECT** to quickly and safely change the selection status of these files, without deleting or changing the files themselves. When searching for files to display in its listing, ***ST*SELECT** recognizes both the normal and the de-selected file extensions. Because the program disk(s) (indicated in the column headings on the ***ST*SELECT** screen) are written to whenever ***ST*SELECT** renames a file, the disk(s) should not be write-protected when ***ST*SELECT** is run.

Since no part of any AUTO folder program or desk accessory file is actually modified by ***ST*SELECT**, you can still change the names or extensions of these files manually. For example, should you accidentally or purposefully disable the ***ST*SELECT** program itself, simply use the *Desktop Show Info...* command to rename it to STSELECT.PRG.

VERIFY

When **VERIFY** is executed from your AUTO folder, the *ST*'s automatic verification of disk writes is disabled. You should notice speed increases in all *ST* programs which write to the disk, with the amount of the increase dependant on the frequency and length of the disk writes.

Chapter 7:

DESK STuff

DESK STuff

This section is for GEM desk accessories. Like AUTO folder programs, they're automatic — they're installed and executed every time your computer is powered up or reset. Unlike AUTO programs, however, desk accessories usually install some part of themselves in the GEM *Desk* menu, where they are available for use from the Desktop, or from within any GEM application which supports the GEM menu system (including **AUTOFOLD** and **FILELOCK**, in the **GEM STuff** section). Some common desk accessory programs include appointment calendars and multi-function desktop tools (such as CORNERMAN, or the *Control Panel* accessory included with your *ST*).

In this section, you'll find the following GEM desk accessory program:

AUTOGEM: At long last, the impossible can be done: with **AUTOGEM** installed in your system, you can now automatically boot a GEM application program. **AUTOGEM** uses its complete knowledge of the GEM Desktop to take control of your mouse, duplicating automatically the exact steps you would use to run a GEM application manually. **AUTOGEM** will work with any *ST* system.

Installing AUTOGEM

To install **AUTOGEM** on your *ST* system, copy the files AUTOGEM.ACC and AUTOGEM.DAT from the **STuff** distribution disk to the root directory (that is, not inside any folders) of the disk you use in Drive A to boot up your system, or to the root directory of your hard drive (if you use one). Both of these files must be present on the disk when you start up your computer.

Whenever the AUTOGEM.ACC and DAT files are present when the system is rebooted, **AUTOGEM** will be installed as a desk accessory. To verify the installation, a box containing the copyright notice for the program will be displayed briefly on the screen during the initialization of the GEM Desktop. To read this notice at greater length, hold down either SHIFT key while the box is being drawn. Release the key to continue the initialization process.

If you are using an older version of TOS which requires that desk accessories have names such as DESK1.ACC and DESK2.ACC, you will need to rename the AUTOGEM.ACC file. Do *not* rename the AUTOGEM.DAT file, for any reason.

> *NOTE:* **AUTOGEM** *may not work correctly with some older versions of the disk-based TOS.*

Using AUTOGEM

AUTOGEM uses an initial 3-second timeout. In other words, if the mouse is still within approximately 1 inch of its initial position (the center of the Desktop screen) 3 seconds after the AUTOGEM copyright notice has been displayed, any program selected and enabled with the AUTOGEM Editor will be autoexecuted. To abort autoexecution, move the mouse at least 1 inch away from its central position, and keep it there until the message Autoexecution Canceled appears in the center of the screen.

To select and de-select programs for autoexecution, use the AUTOGEM Editor.

The AUTOGEM Editor

Selecting the *AUTOGEM Editor* item on the GEM Desk menu will activate the *AUTOGEM Editor* from anywhere within the GEM system. If you are using a floppy-based system, you will need to have the disk you use in Drive A when you boot your system in that drive in order to make any changes with the editor.

When the *Editor* is selected, the *Editor* dialog box will appear on the screen. The top line of the *AUTOGEM Editor* dialog shows the complete path name of the program currently selected for autoexecution (if any). The second line shows the status of that selection (*Enabled* or *Disabled*).

At the bottom of the *AUTOGEM Editor* dialog are four buttons, which control **AUTOGEM**'s functions. They are:

Enable: This button enables autoexecution; in other words, the program indicated above will be automatically executed whenever your computer is reset or powered up. If no program has been selected, or if autoexecution is already enabled, this button will be disabled.

Disable: This button disables autoexecution; when *Disable* has been selected, no autoexecution will occur when the computer is reset or powered up. If autoexecution has already been disabled, this button will also be disabled.

Change: This button allows you to change the program currently selected for autoexecution. When the *GEM Item Selector* appears on the screen, use it to select the program you wish to autoexecute when the computer is reset or powered up. If a valid selection is made, autoexecution of the program selected will be enabled.

Cancel: This button removes the AUTOGEM editor from the screen, without changing either the program currently selected or the autoexecute status. You may also cancel the *AUTOGEM Editor* by pressing RETURN.

If any changes have been made to the autoexecute program or status, the message Writing Data File will appear on the screen, and the AUTOGEM.DAT file will be written to the disk.

Technical Information

AUTOGEM autoexecutes programs by taking over control of the mouse, and by automatically determining and making the same exact movements you would make to select the program desired.

First, **AUTOGEM** opens a new Desktop window for the drive containing the program to be autoexecuted. If all of the Desktop windows are currently in use, **AUTOGEM** uses the *Close Window* menu item to close the top window to make it available.

Next, **AUTOGEM** checks to make sure that the Desktop files are not sorted by date. Due to inconsistencies in the order in which files sorted by date are displayed in the Desktop window, **AUTOGEM** will use the *Sort by Name* menu item to change the way the files are sorted on the Desktop window if *Sort by Date* is currently selected.

If the program is not in the root directory of the drive, **AUTOGEM** uses the mouse to move the window sliders and select the folders to find the program.

Finally, **AUTOGEM** double-clicks on the autoexecute program selected, and returns control of the mouse to you. The program will then be run, just as if you had "walked the windows" and selected it yourself.

Since **AUTOGEM** merely double-clicks on the selected autoexecute item, it can easily be used to load a data file into a program which recognizes certain data file types.

Example:

If your copy of **MI-TERM** is installed on your Desktop (using the *Install Application...* menu item) to recognize document type MIT, telling **AUTOGEM** to double-click on a file with the extension MIT would automatically load and execute **MI-TERM**, which would then automatically load and execute the MIT file.

Error Messages

Certain problems that may occur while **AUTOGEM** is initializing or running will cause an alert to appear on the screen with an error message. Here are further explanations of those error messages:

There is no desk accessory slot available for the *AUTOGEM Editor* — all of the desk accessories slots on the *Desk* menu are in use. Delete or deselect another desk accessory program before attempting to install **AUTOGEM**.

xxxx Error on AUTOGEM Data File xxxxxxxx.xxx
or
xxxxxxxx.xxx Is Not an AUTOGEM Data File

There was a disk-related error reading the data file. The file may have been corrupted by another program, or by a disk error (hardware or medium). Click *ABORT* to remove the alert, and restore the file from a backup copy.

Seek Unsuccessful — xxxxxxxx.xxx Not Found

A folder or file specified in the Autoexecution program pathname is no longer on the disk. Change the Autoexecute program, or restore the file, and reboot.

Unable to Autoexecute —
PRESS ANY KEY TO CONTINUE

AUTOGEM was unable to autoexecute the selected program, for one of these reasons:

DESKTOP.INF in Different Resolution

The current DESKTOP.INF file was saved from a different screen resolution then is currently in use. Resave the Desktop (using the *Save Desktop* command on the *OPTIONS* menu), and reboot.

Window Partially Off Screen

Part of the window **AUTOGEM** would use to autoexecute the selected program overlaps the screen area. Move or resize the top window (if all windows are being used) or the next window, then resave the Desktop, and reboot.

Drive Icon Not Found

There is no drive icon available for the drive specified in the autoexecution pathname. Either change the autoexecute selection to reflect a correct drive; or create a drive icon for the drive using the *Install Disk Drive...* item on the *OPTIONS* menu, resave the Desktop, and reboot.

Window Overlapping Drive Icon

An open Desktop window, or the window **AUTOGEM** would use to autoexecute the selected program, overlaps the drive icon for the drive specified in the autoexecution pathname. Move or resize the window, or move the icon; then, resave the Desktop, and reboot.

Overlapping Drive/Trash Icons

Another icon (another drive, or the trash can) overlaps the icon for the drive specified in the autoexecution pathname. Move the icon, resave the Desktop, and reboot.

If any other messages appear while running **AUTOGEM**, they are messages from the GEM system itself — consult your GEM manuals for further information.

Chapter 8:

GEM STuff

GEM STuff

This section contains GEM application programs. GEM applications (with the extension .PRG) are designed to take full advantage of the built-in GEM system, including the mouse, the drop-down menus, and various other GEM standard features (like dialog boxes and the *GEM Item Selector*). Since they support the GEM menu system, GEM applications also allow you full access to any GEM desk accessories you have installed in your system.

In this section, you'll find the following GEM application programs:

AUTOFOLD: Anyone with more than one program in their AUTO folder will appreciate this program. **AUTOFOLD** makes the task of reordering the files in your AUTO folder (to change the order in which they execute) quick and easy.

FILELOCK: This program uses a proprietary algorithm developed by FACTORY PROGRAMMING to keep your files a secret. Using up to three different passwords for each "locking" or "unlocking" operation, **FILELOCK** can provide many different levels of security for confidential or sensitive information.

Both of these programs will work with any *ST* system.

Installing GEM STuff

To install **GEM STuff** programs on your *ST* system, copy each program file you want to install (all of which have the extension .PRG) from the distribution disk to a formatted data disk. This does *not* need to be the same disk you use to boot your computer.

To use a **GEM STuff** program, simply double-click on the name or icon of the program on your Desktop, or type RUN, followed by the name of the program, from the **DOS Shell**.

GEM STuff Programs

AUTOFOLD

When you run **AUTOFOLD**, the screen will clear, and you will see a window in the center of the screen, containing twenty program name slots, in two columns of ten each. The twenty slots show the first twenty programs in the AUTO folder on the default drive, if there is an AUTO folder on that drive.

To change the order of the files in the **AUTOFOLD** window, use the mouse to grab a file by pointing at its name and holding down the mouse button. Then, drag the file to its desired location by moving the mouse (with the button still depressed) until the moving outline corresponds to the new location, and then releasing the button.

You will also see three menus on the **AUTOFOLD** screen. The **AUTOFOLD** menu allows you to display the copyright notice, or to use any desk accessories your may have installed in your Desktop.

The *FILE* menu gives you these choices:

Reorder: Rearranges the files in your AUTO folder to match the arrangement shown in the AUTOFOLD window. If the programs in the AUTO folder are already in the order shown, this menu item (and the matching REORDER button in the AUTOFOLD window, which performs the same function) will be disabled.

Quit: Exits the **AUTOFOLD** program. You can also exit **AUTOFOLD** by clicking on the matching *Quit* button in the **AUTOFOLD** window, or by pressing CONTROL/UNDO.

The *Drive* menu allows you to select which drive your AUTO folder is located on, with the default being the drive where AUTOFOLD is located. The drive currently selected is indicated with a check mark.

Technical Information

AUTOFOLD works by copying all of your AUTO folder files into memory, deleting the original files from your AUTO folder, and then rewriting them back to the disk in the selected order. You can watch the file names in the AUTOFOLD window for a visual representation of this process as it occurs.

This method of reordering the files was chosen as being safer than quicker methods which involve directly writing to the disk directory, as a disk error or computer glitch in this type of program could result in the loss of all data on the disk.

Error Messages

Certain problems that may occur while AUTOFOLD is initializing or running will cause an alert to appear on the screen with an error message. Here are further explanations of those error messages:

AUTOFOLD Window Could Not Be Opened

There were no windows available to GEM. This usually indicates that a previously run program has tampered with memory that did not belong to it. Reboot your computer, and try again.

Cannot Allocate Enough Memory for Screen Buffer

There is not enough memory available for AUTOFOLD to run. Increase your memory, or delete or deselect unused desk accessories or AUTO folder programs, and try again.

Cannot Allocate Enough Memory for File Reordering

There is not enough memory available for AUTOFOLD to load all of your AUTO folder files. Increase your memory, or delete or deselect unused desk accessories or AUTO folder programs, and try again.

Disk Error xxxxxxx File xxxxxxxx.xxx

There was a disk-related error involving the specified file. The file may have been corrupted by another program, or by a disk error (hardware or medium). Click *Retry* to attempt the operation again, or click *Abort* to remove the alert, and restore the file from a backup copy.

If any other messages appear while running **AUTOFOLD**, they are messages from the GEM system itself — consult your GEM manuals for further information.

FILELOCK

When you run **FILELOCK**, the screen will clear, and you will see two menus on the screen. The **FILELOCK** menu allows you to display the copyright notice, or to use any desk accessories your may have installed in your Desktop. The *File* menu gives you these choices:

Lock File...:	Locks (encrypts) a selected file
Unlock File...:	Unlocks (decrypts) a selected file
Quit:	Exits the FILELOCK program. You can also exit FILELOCK by pressing CONTROL/UNDO.

When you select either *Lock File...* or *Unlock File...*, you must then select the file to be encrypted or decrypted, using the standard *GEM Item Selector*. When the file has been selected, you will see the lock/unlock dialog on the screen.

To lock a file, enter the password(s) you wish to use, and click *OK*, or press RETURN. Click *Cancel* to abort the *Lock* command.

To unlock a file, you must enter the password(s) that were used to lock the file, exactly as they were entered when the file was locked. Click *OK*, or press RETURN. Click *Cancel* to abort the unlock command.

The slider bar on the screen shows you the progress of the locking or unlocking process. When the command is completed, the screen clears again. You may now select another command, or exit **FILELOCK**.

Technical Information

FILELOCK uses a proprietary algorithm developed at FACTORY PROGRAMMING to encrypt and decrypt files. The complexity of the scrambling technique used depends directly on the number and length of the passwords entered. If no passwords are used, a simple bit-reversal method is used, which, while sufficient to shield the file's contents from the novice user, is rather easily broken by a more experienced cryptographer. The best results are obtained by using all three passwords, using longer passwords, and by using password of different lengths. The repeat cycle of the algorithm can be calculated as the "lowest common multiplier" (LCM) of the lengths of the passwords used. The best possible encryption, using three passwords of 28, 29, and 30 characters, repeats itself after every 24,360 bytes. Since the number of passwords used and the passwords themselves are not stored in the encrypted file in any form, this variable-length encryption is extremely difficult to break.

For additional security, is it always possible to lock a file with one set of passwords, and then lock it additional times with the same or different passwords. To decrypt such a multiply-locked file, unlock the file using the last set of passwords first, and then continuing in reverse order.

Since locked files are not copy-protected in any way, they can be easily transported from one machine to another, or sent by modem to another computer using any standard 8-bit communications channel (an error-checking protocol, such as XMODEM, is recommended). An IBM-compatible version of **FILELOCK** (identical to and completely compatible with the *ST* version) is available from the Factory by special order.

As part of its encryption and decryption process, the file you select to lock or unlock is overwritten and destroyed by the newly locked or unlocked file. There are two reasons for this: first, it allows you to lock or unlock a file regardless of available free disk space; and second, it enhances security, as nothing invalidates an encryption process as quickly as leaving a "plain text" copy of an encrypted file on your disk. However, this can lead to problems if decrypting passwords are entered incorrectly, as the improperly decrypted file will then be completely unusable. When in doubt about a password, make a backup copy of the locked file before attempting to unlock it.

As with all password protection systems, certain rules of thumb apply. For maximum protection, you should not leave your password(s) lying on your desk, let others see your password(s) as you enter them, or send them as open messages over public channels. Finally, ***please*** **do not forget your password(s)** — even we can't decrypt a file without them. When in doubt, make a backup copy of a locked file first before trying to decrypt it.

Error Messages

Certain problems that may occur while **FILELOCK** is initializing or running will cause an alert to appear on the screen with an error message. Here are further explanations of those error messages:

FILELOCK Window Could Not Be Opened

There were no windows available to GEM. This usually indicates that a previously run program has tampered with memory that did not belong to it. Reboot your computer, and try again.

Cannot Allocate Enough Memory for Screen Buffer

There is not enough memory available for **FILELOCK** to run. Increase you memory, or remove unused desk accessories or AUTO programs, and try again.

xxxx Error on Data File xxxxxxxx.xxx

There was a disk-related error reading the specified file. The file may have been corrupted by another program, or by a disk error (hardware or medium). Click *Abort* to remove the alert, and restore the file from a backup copy.

If any other messages appear while running FILELOCK, they are messages from the GEM system itself; consult your GEM manuals for further information.

Chapter 9:

TOS STuff

TOS STuff

This section contains TOS programs. Unlike GEM applications, TOS programs (with the extension TOS) do not use the GEM interface, relying instead on the built-in VT-52 emulation for their screen output (which is usually minimal and non-interactive). By definition, TOS programs do not require or accept parameters from the calling program (the Desktop or the **DOS Shell**). TOS programs can still be very useful, however; for example, AUTO folder programs (see **AUTO STuff**) are actually TOS programs, forced to wear a false PRG extension.

In this section, you'll find the following TOS programs:

512K: This program electronically "unhooks" your brand-new 1-meg upgrade, or turns your *1040ST* or *Mega-ST* into a *520ST*. **512K** compensates for the poor programming and copy protection of programs which can't operate properly with more than 512K of memory available.

KEYCODE: This program returns the TOS scan code for any selected key. Programmers can use **KEYCODE** as a development tool; non-programmers will find it useful with FORMFEED (see the **AUTO STuff** section).

Both of these programs will work with any *ST* system (although using **512K** on a machine with only 512K of memory to begin with won't accomplish much).

Installing TOS STuff

To install **TOS STuff** programs on your *ST* system, copy each program file you want to use (all of which have the extension TOS) from the distribution disk to a formatted data disk. This does *not* need to be the same disk you use to boot your computer.

To use a **TOS STuff** program, simply double-click on the name or icon of the program on your Desktop, or enter the name of the program from the **DOS Shell**.

TOS STuff Programs

512K

When you run **512K**, the screen will clear, and the message Booting in 512K mode will be displayed. After a 5-second pause (to allow you to change disks, if necessary), your *ST* will reboot as a 512K machine.

Even if your *ST* is rebooted again, it will remain in **512K** mode. To return to full memory access, use the **RESET** program before rebooting, or turn off the power switch on your *ST*, wait approximately 10 seconds, and restore power.

KEYCODE

When you run **KEYCODE**, the screen will clear. Pressing any key will display the full 24-bit scan code used by the *ST* to represent that key (or combination of keys, with ALTERNATE, CONTROL, and/or SHIFT).

Scan codes will continue to be displayed for all keys until the RETURN key is pressed. Pressing another key (after the scan code for RETURN is displayed) exits the KEYCODE program.

Chapter 10:

TTP STuff

TTP STuff

This section contains TTP programs. Programs with the TTP extension (which stands for "TOS — takes parameters"), like TOS programs, do not use the GEM interface, but use the VT-52 emulation for their screen output. Unlike TOS programs, however, TTP programs require one or more parameters of some kind (such as a file name or command) in order to function properly. TTP programs are usually used to perform some operation (like printing, sorting or editing) on one or more selected disk files.

In this section, you'll find the following TTP programs:

FC: This program compares two program or binary data files, and shows the bytes which differ between them. FC can be used to verify copied files, or to compare different versions of the same program.

FDEL: Unlike a standard *Delete* command (such a dragging a file to the trash can on the Desktop), which merely erases a file's directory entry, **FDEL** erases the actual disk sectors which contain the file, as well as removing the directory entry. For files containing confidential or sensitive data, **FDEL** protects your deleted data from unauthorized recovery.

GREP: This is a text search program. **GREP** (which stands for "Generalized Regular Expression Parser") allows you to search one or more text files for any given string. Numerous options allow you to easily find any desired string in any group of files.

HEADER: This program shows you some of the "technical" data about a program file — namely, the sizes of the various segments of the program, and the length of the symbol table (if one is present). **HEADER** is a useful and informative tool for the *ST* programmer.

HEX: This program display the contents of a disk file in the "base language" of a computer — hexadecimal, or base 16. Another useful programmer's tool.

TOUCH: This program literally "touches" disk files, changing their dates and times to the current system date and time, or to a specified time. **TOUCH** can be used in conjunction with other programs which are dependant on file dates and times (such as "make" utilities or hard disk backup programs), or to "pretty up" a disk full of files for distribution.

UNHIDE: This program allows you to change the attributes of any *ST* disk file. The "hidden", "system" and "read only" flags can be easily altered for any file or group of files.

All of these programs will work with any *ST* system.

Installing TTP STuff

To install **TTP STuff** programs on your *ST* system, copy each program file you want to use (all of which have the extension TTP) from the distribution disk to a formatted data disk. This does *not* need to be the same disk you use to boot your computer.

To use a **TTP STuff** program from the GEM Desktop, double-click on the name or icon of the program on your Desktop. Enter the parameters required for the program in the *Open Application* dialog, and click *OK*, or press RETURN, to run the program.

To use a **TTP STuff** program from the **DOS Shell**, simply type the program name and parameters, separated by one or more spaces.

Paths

All of the **TTP STuff** programs accept file names in a particular format, called a *path.* The format of a TOS path name is:

[<drive>:][\][<folder>][\<folder...>][\][<file>][.<ext>]

Drive: A letter from A through P (or a through p), designating the disk drive for the file or path. When a drive is specified, the drive letter must be followed by a colon (:). When no drive is specified, the current drive is used.

Folder: A valid TOS folder name. When more than one folder name is used, they must be separated by the backslash character (\). If the first folder specified is preceded by a backslash, the path starts at the root directory of the drive; otherwise, any folders specified are relative to the current folder. If no folder is specified, the current folder is used.

File, ext: A valid TOS file name (up to 8 characters), with an optional extension (up to 3 characters). If an extension is used, a period (.) must separate the file name and extension. If a folder has been specified, a backslash (\) must separate the last folder name and the file name. If no file or extension is specified, all files in the selected folder are used.

Examples of Valid File/Pathnames:

FILE NAME.EXT: Specifies the file FILE NAME, with the extension EXT, in the current folder on the current drive.

FOLDER\FILE NAME.EXT: Specifies the file FILE NAME, with the extension EXT, in the folder FOLDER, which is contained in the current folder on the current drive.

A:FILE NAME.EXT: Specifies the file FILE NAME, with the extension EXT, in the current folder of Drive A.

D:\FOLDER\FILE NAME.EXT: Specifies the file FILE NAME, with the extension EXT, in the folder FOLDER on drive D.

Some of the **TTP STuff** programs require or allow more than one path in the parameter.

Wildcards

Some of the **TTP STuff** programs accept special characters, known as *wildcards*, in path names. Wildcard characters (* and ?) are used in file names and extensions (never in drive or folder names) to specify groups of files (such as all PRG files, or all files whose names begin with the letter C). The ? wildcard is used to match any character in one character position (or no character in that position), while the * wildcard matches any one or more character positions (or no characters at all in that position).

Examples of the Use of Wildcards:

***.PRG**	This specifies all files with the extension PRG.
FILE NAME.*	This specifies all files with the name FILE NAME, with any (or no) extension.
F*.T??	This specifies all files where the first character of the file name is F, and the first character of the extension is T. Using this file specification on the STuff distribution disk would match FC.TTP and FDEL.TTP, but not FORMFEED.PRG,FILELOCK.PRG or any of the other files with the TTP extension.
. or **????????.???**	This specifies all files (in the current folder).

Where wildcards are allowed, and multiple paths are supported, the files selected are all members of one or more of the groups specified. For example, the multiple path *.C *.H would select all files with either the C or H extensions. The multiple path AUTO*.* *.PRG would select all files where the first 4 characters of the file name were AUTO, and all files with the extension .PRG. However, any files matching both of these paths would be acted upon by the program *twice*, once during each "pass".

Switches

Many of the **TTP STuff** programs accept special parameters known as *switches*, which are used to control the operation of the program, or to select certain options. Switches are specified with the hyphen character (-), which is followed immediately by a single letter indicating which switch is being selected.

All switches specified must be at the beginning of the parameter passed to a **TTP STuff** program, before any path name(s) specified, and separated from them by at least one space. If more than one switch is specified, each must consist of both the hyphen character and the switch letter, and must be separated by at least one space from each other.

The switches recognized by the **TTP STuff** programs are:

-c Display only the count of matching lines, rather than the lines themselves. Used by **GREP**.

-d Specify date and time. Used by **TOUCH** (see the **TOUCH** instructions for parameter format).

-f Don't print file names. Used by **GREP**.

-h Set "hidden" attribute of selected file(s). Used by **UNHIDE**.

-n Number the matching lines. Used by **GREP**.

-p Pause until a key is pressed before ending the program. This switch is used when running a **TTP STuff** program from the Desktop, in order to pause the display for viewing. This switch is also activated automatically when one of the **TTP STuff** programs is called without parameters. Used by **FC, FDEL, GREP, HEADER, HEX, TOUCH**, and **UNHIDE.**

-r Recursive Folder Walk. Repeats the command(s) for the files in the specified folder, and the specified files in all folders included in the specified folder. For example, the -r switch, followed by the path *.* (selecting all files in the root directory), would apply the program's operation to all files on the current drive, regardless of folder location. Used by **FDEL, HEX, TOUCH,** and **UNHIDE.**

-s Set "system" attribute of selected file(s). Used by **UNHIDE.**

-v Print non-matching lines. Used by **GREP.**

-w Write protect (set "read only" attribute) selected files. Used by **UNHIDE.**

Where multiple switches are allowed, they may be specified in any order.

TTP STuff Programs

FC

Parameter format:

[-p] PATH1 PATH2

FC compares two binary data files, and displays the addresses (relative to the beginning of the files) where they differ, and the byte values in which they differ.

FC does not accept wildcards.

FDEL

Parameter format:

[-p] [-r] PATHNAMES

FDEL fills selected files with zeros, and then deletes them. The zero fill assures that no trace of the deleted file remains on the disk.

FDEL accepts both wildcards and multiple paths.

> **WARNING:** *Many disk utility programs contain commands to "recover" deleted files using the information still present on the disk after a standard file deletion. Files deleted with* **FDEL** *cannot be recovered by these programs.*

GREP

Parameter format:

[-c] [-f] [-n] [-p] [-v] EXP PATHNAMES

GREP displays all occurrences of the specified ASCII character string (EXP) in the specified file(s). If the string contains spaces or punctuation characters, it must be enclosed with either single or double quotes (' or ").

GREP accepts both wildcards and multiple paths. Multiple search strings are not supported.

The EXPression can contain any or all of the following characters, to activate special **GREP** functions:

^	match beginning of line.
$	match end of line.
.	match any character, except a new line.
:a	match any alphabetic character (A-Z, a-z).
:d	match any digit (0-9).
:n	match any alphanumeric character (A-Z, a-z, 0-9)
:_	match any "white space" character (blank, tab, new line).
*	match any zero or more occurrences of the previous character.
+	match any one or more occurrences of the previous character.
-	previous character optional.
\	match following character exactly (escape code).
[]	match any one of the enclosed characters.

HEADER

Parameter format:

[-p] PATHNAMES

HEADER displays the segment breakdown of the selected program file(s). The lengths of the TEXT, DATA, and BSS segments, as well as the length of the symbol table (if one is present) are displayed.

HEADER accepts multiple paths, but does not accept wildcards.

HEX

Parameter format:

[-p] [-r] PATHNAMES

HEX displays the selected file in hexadecimal format.

HEX accepts both wildcards and multiple paths.

TOUCH

Parameter format:

[-dMMDDYYHHMMSS] [-p] [-r] PATHNAMES

TOUCH sets the date and time of selected file(s) to the current date and time, or the specified date and time (passed in *exactly* the format shown).

TOUCH accepts both wildcards and multiple paths.

UNHIDE

Parameter format:

[-h] [-p] [-r] [-s] [-w] PATHNAMES

UNHIDE changes the file attributes of the selected file(s) to the attribute(s) selected. For example, if the h and w flags are specified, all files in the path(s) given are changed to "hidden" and "read only". If no attribute flag are specified, all attributes are removed from all files in the path(s) given.

UNHIDE accepts both wildcards and multiple paths.

PATCHER.BAS

Included on your distribution disk is **PATCHER.BAS**, an *Atari ST* BASIC program designed to implement short program patches. Should a minor change to any of the programs in your copy of **STuff** ever be required, it may be distributed in the form of a set of parameters to be entered into the **PATCHER** program, allowing you to make corrections to the program without using a complex sector editor or returning your disk to us.

To use **PATCHER**, first run BASIC.PRG from your *ST Languages* disk. Next, insert a disk containing PATCHER.BAS in the drive, and enter the command RUN "PATCHER" at the BASIC prompt to load and execute the **PATCHER** program.

When the screen clears, you will see the **PATCHER** copyright notice, and the first prompt. Throughout the program, whenever a prompt appears on the screen, type in whatever parameter the program is asking for. Use the BACKSPACE key to correct mistakes, and press RETURN to complete the entry. Pressing RETURN without entering a parameter will move you back to the preceding prompt.

These are the **PATCHER** prompts:

File Name: The name of the file you wish to patch. Make sure that program is on the disk in the drive, and type the complete name of the program (for example, AUTOGEM.ACC). If **PATCHER** cannot find the file, an error message will be displayed.

Relative Address: The hexadecimal address of the program byte you wish to modify, with the first byte of the program being numbered 0, the second byte 1, etc. You will see an error message if an invalid address is entered.

Current Value: The hexadecimal value of the current byte at the specified relative address. This value is requested to double-check the relative address entry, avoiding erroneous patches. An error message will be displayed if an invalid or incorrect value is entered.

New Value: The new hexadecimal value for the byte at the specified relative address. An error message will be displayed if an invalid value is entered.

Note: the value of the specified byte is changed immediately, so please enter it carefully.

When a new value is entered and accepted, **PATCHER** returns you to the *Relative* address prompt, to enter the address of the next byte to be changed.

Patches to be used with **PATCHER** will look like this:

```
AUTOGEM.ACC:
  12345  67 -> 89
  1ABCD  EF -> 01
```

This means: "in the file AUTOGEM.ACC, change relative byte 12345 from its current value of 67 to a new value of 89, and then change relative byte 1ABCD from EF to 01".

PATCHER.BAS has been released to the public domain. You are invited to upload the program to bulletin board systems, and to distribute copies of the program to users groups and friends. However, you are *not* allowed to sell the program, or to charge anyone else for it, or for its use or copying.

Part III

Superdirectory

A Powerful, Easy to Use

Disk Cataloging Program

By Mark Feldman

Chapter 11:

The Essentials of Superdirectory

Introduction

Superdirectory is a powerful, easy to use disk cataloging program, which runs under the GEM operating system. It allows you to keep track of all your floppy and hard disk files in a very convenient format.

Terms and Definitions

Some terms used in the documentation:

Record: A single disk file entry. For example, a record consists of:

Filename
Disk number
Category
Filesize
Pathname
Remarks

Note: Each Record *is composed of* Fields.

Field: A subset of a record. All of the following are fields: filename, disk number, category, remark, pathname, etc.

Data File: This refers to the records currently in memory or a *Superdirectory* data file on disk.

Subset: *Subset* means exactly that, a subset of a set of items, letters, files, records, etc. This is analogous to having a set of letters { A B C D}, then { A C } would be a considered a subset of this, but { E F } would not. By using the *All* command in *FIND* you generate a subset data file of your main data file.

Category: This is a one character field. It allows you to designate a category belonging to each record. For example, all records that are utilities can have the category "u" for utility, or "g" for games, "b" for business, etc. You can search and sort your data file based on this field. Using the category field frees you to use your remarks for more elaborate descriptions. See remark definition below.

Remark: Each record can contain a remark (or comment) up to 25 characters long. This can be used to identify what each record contains or what it's function is. For example: "Good text editor", "Backup copy", "Working version", etc. Remarks can be sorted and searched.

Commands

ADD

This command is used to add a disk to the **Superdirectory** data file.

You must enter a disk number of up to 3 characters. We suggest using only numeric characters. For example: 001, 002, etc., and not abc, 23a, etc, although both are allowed.

Disk Number:	Enter a 1 to 3 character disk number.
Drive Selection:	Type in the desired drive number from the keyboard.
Number : of Records	This tells how many free records are left in memory.

OK: Begin ADDing disk.

Cancel: Exit back to the main screen.

If you are out of memory, the program will warn you of the condition and then continue to process the disk. The program handles running out of memory in the following manner: it will continue to read the entire disk, deleting any inactive subdirectories. If this opens up memory, then more files will then be added. The important point is that if an out of memory message appears during the *ADD* operation, you should *ADD* the same disk again. This will allow a more complete update of all the files, since more

memory may have opened up during the previous *ADD*. At that time, you should strongly consider starting a new data file. See the *All* command in *FIND* for directions on how to easily break up your data file into smaller modules.

> *Note: Before reading a disk,* **Superdirectory** *automatically sorts your data file by disk number, if this has not been done already.*

> *Note: If you are in the* FIND/All *mode, the program automatically exits this mode before ADDing the disk. See the* FIND *command for more details on the* FIND/All *mode.*

SEARCH

The whole purpose of **Superdirectory** is to make it easy to locate files in your collection. **Superdirectory** uses a very powerful feature called a "mask" or "filter" to search through the data file. A mask is a template that you define. The search is made by attempting to match a record with the mask. Wildcards and dummy characters can be used.

For example, if you are looking for all the records with the filename *CHESS.EXE*, then just enter *CHESS.EXE* under the *Filename* field and click on *OK*. The program will put the first matching record at the top of the screen in reverse video. You may also fill in the other template fields. For example, if you are looking for the records *CHESS.EXE* that are only on subdirectories named *\GAMES*, then enter *\GAMES'* in the *Pathname* field and click on *OK* or *All*. You are allowed the use of any combination in order to effectively search through your records. You can continue the search for the next matching record by clicking on the *NEXT-FIND* command button in the main menu. Also see the *FIND/All* mode and *SEARCH* commands.

Wildcards and Dummies:

Using wildcards ("*") and dummy characters ("?") greatly enhances the usefulness of the *FIND* command. "*" tells the program that any matching string up to the occurrence of the "*" will be a match. Thus, CHE*.*, C*.*, and *.EXE all would match with the filename *CHESS.EXE* (similar to how wildcards work with the *DIR* command in DOS).

Dummy character ("?") works as an automatic match. For example: *CH?SS.EXE* and *?HE?S.EXE* matches *CHESS.EXE*, but *?HEST.COM* does not, although *?H*.EXE* does.

> ***Note**: Empty fields are ignored.*
>
> ***Note**: The matching record will be put in reverse video at the top of the screen.*
>
> ***Note**: When searching for a record you know exists in the data file and the program says that there are no matches, use the sliders to go to the top of the data file, and then search for it again. Chances are you began past the matching record in the data file and simply needed to go to the beginning of the file in order to find the matching record(s). This is not necessary in the* FIND/All *mode which automatically begins the search from the beginning of the data file.*

OK: Puts first matching record at the top of the main menu screen.

All: Puts the program into the *FIND/All* mode. The *FIND/All* mode works as follows: once you have set the *SEARCH* mask, clicking on *All* tells the program to find every matching record, and to make a subset data file of these matches. The program will now work exactly as before except:

- The files being displayed are only the ones that match the *SEARCH* mask.

- When you try to ADD a file while in the FIND/All mode the program automatically exits this mode before ADDing the disk.

- When you try to *Save* this data subset file to disk, a warning appears to indicate that you are not saving the entire data file but simply a subset of it. Consequently, you will want to give it a different name than the current one; otherwise you will overwrite the data file on the disk with only a subset of the data file.

  ***Note**: The program will automatically reinstate the original data file filename when you exit from the* FIND/All *mode.*

The *FIND/All* mode is a very useful command. It allows you to:

- Break up your entire data file into subsets that you can save to disk. For example, let's say you want a data file with only your *.EXE records; you can set the *SEARCH* mask , click on ALL, and then go to *SAVE* and these records will be saved to disk under the data file name that you choose.

  Note: *When you exit the* FIND/All *mode the original filename is reinstated.*

- Break up the data file into more manageable chunks that fit into your system's memory when it gets too big.

- Total the number of matching records. For example, by setting your *SEARCH* mask for *.EXE filenames, this application will give the total number of *.EXE records in your data file at the top of the main menu screen.

- Conveniently group together the records you want to work on so you can FIND, EDIT, SORT, etc. more quickly and easily.

  Note: *All* **Superdirectory** *commands, except FIND, work while in the* FIND/All *mode.*

Clear: Clears all the *SEARCH* mask fields.

Reset: Resets *SEARCH* mask to original settings.

Cancel: Exit back to main menu.

LOAD

This loads specified data file from the disk into memory. If there are records currently in memory, this command does not work.

This allows the merging of files, saving the most recent version in memory. However, it will not overwrite any of the macros already in memory. All remarks and categories are retained in memory.

PRINT

The mask in this command works almost identical to the *SEARCH* mask. The difference is that ALL the matching records go to the printer and not the screen. Fill in the *PRINT* mask based on which records you want printed. For example, if you only want to print out records with the disk number 001, then enter 001 into the disk number field and click on *Mask*. Empty fields are ignored.

Cancel: Exit back to main menu.

Window +: Print all of the data file starting from the top of the current screen.

> ***Note***: *To print an entire data file you must go to the top of the data file using the sliders. If the data you need printed does not appear at the beginning of the screen it will not be printed. Only the data that appears at the top of the screen, and that which follows, will be printed.*

Note: Format of the printout is based on the PRINT/Format *command settings. See the* FORMAT *command.*

Clear: Clears all *Print/Mask* fields.

Enter: Same as *OK* command except that if no matches are found, the program remains in *Print* mode instead of going back to main menu.

OK: Prints all the records that match the *Print* mask.

Reset: Resets *Print* mask fields to original settings.

Window: Prints records currently on screen. The format of printouts are based on the *Print/Format* command settings.

Format: Utilizes *Print* format dialog which offers a choice of five different print formats (because standard printers usually support only 80 characters per line, certain compromises were made in order to work within these constraints).

The PRINT/Format *Options:*

Full Pathnames: Prints filename, disk number, category, date, filesize, and full pathnames (no remarks).

Full Remarks: Prints filename, disk number, file size, date, and full remarks (no pathnames).

Filenames: Three across; prints 3 records per line - filename, disk number, and filesize.

Pathname: Two across; prints 2 records per line - filename, disk number, date, and first 11 characters of pathname.

Remark: Two across; prints 2 records per line - filename, disk number, date, and first 11 characters of remark.

***Note**: If your printer is not on-line, the program may take several seconds before going back to the main menu. It is NOT locked up.*

***Note** : The format of the printout is always based on the* PRINT/Format *command settings.*

***Note**: Pressing the SHIFT key during printing will cause an abort back to the main menu.*

SAVE

Save a specified data file from memory to disk. If you are in the *FIND/All* mode you will be warned that you are saving only a subset of your entire data file to disk. See *All* in the *FIND* command.

***Note**: Each record requires 80 bytes of disk space. So if you are saving 1000 records, you must have at least 80000 bytes of free disk space in order to save the data file to disk.*

SORT

You can sort records by category, disk number, extension, filename, pathname, or remark.

This sorts records of all nine fields found under the *SORT* option.

Note: The disk number SORT *also subsorts the records by filename and pathname.*

Cancel: Exit back to main menu.

All: *SORT* will ignore disk number and sort entire data file based on the field chosen for sorting.

Disk: Specifies sort only within each disk number. For example, if you sort by filename with the *All* option, your entire data file will be sorted by filenames only. If you sorted by filename with the *Disk* option, then filenames are sorted alphabetically only within each disk, and not within the entire data file.

Note: The All *and the* Disk SORT *options can be a bit confusing at first. Experimenting with these two* SORT *options, by sorting one of your initial data files, will help you quickly understand the difference.*

DISK

This finds the next disk in the data file. Allows you to step through records by disk number. This is especially useful when records are sorted by disk number. This mode allows you to run a continuous loop when stepping through the data files. When the end of the data file is reached the search will resume at the beginning. Thus, no matter where you begin your search **Superdirectory** will find the desired disk.

SEARCH

Finds the next occurrence of matching records using the *SEARCH* mask; remember the *SEARCH* mask must be set first. See the *FIND* command.

> *Note: The matching record will be put in reverse video at the top of the screen.*
>
> *Note: The* SEARCH *mask also forms a continuous loop, so if a matching record exists* **Superdirectory** *will find it.*

PATH

This finds the next pathname in data file and allows you to step through data file by pathname. This function is more useful if your data file is sorted by pathname. The continuous loop is again evident in this mode.

DRIVE

Clicking on the drive selection results in an appearance of a dialog from which the current drive selection is chosen.

EDITOR

This allows the user to edit records. Enter into the *EDIT* mode by clicking on one of the records being displayed on the screen. You can now directly alter the contents of the record. A *Delete File* option is accessed through this mode. You are able to add remarks and edit macros. Any new macros and remarks are saved along with the data file.

Window+: Allows you to *Edit* all the records starting from the top of the screen to the end of the data file.

> ***Note**: You must use the sliders to go to top of the data file in order to start editing from the beginning of the data file.*

Window: Allows you to *Edit* all the records on the screen individually.

Reset: Reset fields to original settings.

Cancel: Exit back to main menu.

Defaults: In this dialog there are 12 standard remark entries. Clicking on one of these options will automatically enter it into the remark field of the record you are editing. Otherwise, you can enter your own remark.

> ***Note**: If you alter the field on which the data file is sorted, then the program will sort the data file upon exiting from the* EDIT *mode. This is done to maintain the integrity of the sorted field.*

In the title (boot-up) menu, clicking on *FILES* brings up a window with three options:

Superdirectory: Enter into main program

Clear: Erase current data file and re-initialize program.

Exit: Exit to TOS

Note: If there are already records in memory, then a warning dialog will appear reminding you to save the data file before exiting or clearing the program. Remember, in order to permanently save your data file, you must always save the data file to disk. See the SAVE *command.*

Part IV

M-Disk *Plus*

Two Programs that Reserve Portions

of Your Computer's Memory

To Load and Save Data

And To Allow Work While Printing

By Timothy Purves

Chapter 12:

The Essentials of M-Disk Plus

Introduction

M-Disk ***Plus*** is a combination of two great programs from MICHTRON. **M-Disk** and **Soft Spool** were joined, and vastly improved, to give the user one fabulous software package!

M-Disk uses a portion of your computer's memory to load, and save data, as if it were a normal disk drive. The "phantom", or RAM Disk drive, is faster and more reliable than a hardware drive, because it has no mechanical parts.

With **Soft-Spool**, you and your printer can now work at the same time. **Soft-Spool** works by reserving a section of your computer's memory as a print buffer. Any data to be printed, is stored in the **Soft-Spool** buffer, then transferred to the printer as the printer becomes ready, without interrupting other operations of the computer.

Configuring M-Disk *Plus*

The program CONFIG.TOS is used to set the memory usage of **M-Disk** and **Soft-Spool**. To execute the program double click on the icon CONFIG.TOS. The screen will clear and show three options.

1. Config MDISK
2. Config SPOOL
3. Exit Program

When Option 1 is selected, the configuration program will attempt to open the file MDISK.PRG. If it cannot find it, an error will be displayed. If the program is successful, you will be prompted which drive for M-DISK (D-P). Enter the desired drive letter. Next you will be prompted:

Memory to be allocated for M-DISK:

Valid sizes are from 1-800 (pressing return will allocate the maximum memory possible); these values are then saved back to disk.

When Option 2 is selected, the configuration program will attempt to open the file SPOOL.PRG. If it cannot find it, an error will be displayed. If the program is successful, you will be prompted:

Memory to be allocated for spooler:

Valid sizes are from 1-800 (pressing return will allocate the maximum memory possible), and these values are then saved back to disk.

When Option 3 is selected it will return to the DESKTOP.

How to Install M-Disk *Plus*

You must install **M-Disk *Plus*** before executing any program that uses them. This can be done automatically, at boot time, or manually. To reclaim the memory that **M-Disk *Plus*** has reserved, you must reboot the computer. **M-Disk *Plus*** will not install if the memory usage specified would leave less than 128 kilobytes of memory .

Manual Installation

Double-click on the icon labeled MDISK.PRG or SPOOL.PRG for the program title screen to appear. If the program cannot install itself, an error message will appear and wait for you to press RETURN to continue. Refer to the section on error messages for an explanation of the errors that may occur.

Automatic Installation

Drag the file(s) MDISK.PRG/SPOOL.PRG to an AUTO folder on your boot disk. To create an AUTO folder refer to your owners manual. Now, whenever you reboot, the programs in the AUTO folder will be automatically loaded. To abort the loading of **M-Disk *Plus***, hold down the CONTROL, SHIFT, or ALT keys.

How to Use M-Disk *Plus*

At the Desktop, select a drive icon, then choose *Install Disk Drive* from the *OPTIONS* menu. Type the drive name set in CONFIG and RAMDISK, as the drive identifier. Click on *Install*, then select *Save Desktop*. This will save the drive icons in the DESKTOP.INF file, and will cause the *RAMDISK* icon to appear automatically.

How to Use Soft-Spool

Once installed, **Soft-Spool** invisibly intercepts BIOS print calls. You don't need to do anything special to use it, just make print requests as normal. When non-BIOS print calls are made, **Soft-Spool** is ineffective, but will not interfere with the print operation. **Soft-Spool** can send data to the printer at speeds of up to 200 characters per second.

Error Messages

Can't Allocate Memory for M-Disk

There wasn't enough memory to create a ram disk and still leave 128 kilobytes for TOS.

Can't Allocate Memory for Spooling

There wasn't enough memory to create a *Soft-Spool* buffer, yet still leave 128 kilobytes for TOS.

SPOOLER Already Active
Clear Backlog (Y/N) ?

Soft-Spool is already installed. You may clear all data waiting in the printer queue by responding with Y.

Part V

DOS Shell

A Program that Mimics MS-DOS™

Commands on the GEM-based

Atari ST *Computer.*

By Timothy Purves

Chapter 13:

An Introduction to DOS Shell

Introduction

MS-DOS ("MicroSoft™ Disk Operating System") is a powerful, popular disk operating system used on many home and business computers. Its speed and organization have made it the de facto king of the operating systems in the business world.

Atari chose to go with a GEM operating system because the interface is more user friendly. If you are already comfortable with DOS or want to learn and enjoy its speed and power, **DOS Shell** will let you mimic typical MS-DOS commands on your *Atari ST* computer.

The **DOS Shell** commands can list which files are on a disk, check the free space left, copy files and more, much more quickly than GEM mouse operations. Global filenames make handling more than one file at once much easier from **DOS Shell** than from GEM.

Instead of making you point at and drag icons and windows, **DOS Shell** lets you use short English commands to control your disk drives. You can even use command abbreviations to save on typing.

When you run **DOS Shell**, the normal GEM desktop display is replaced by a clear screen with a prompt and a blinking cursor.

If you type one of the DOS commands and press RETURN, **DOS Shell** will execute the command. You'll be surprised how quickly you can handle everyday disk operations. And **DOS Shell** is ideal for handling your hard disk.

Loading Instructions

Before you run **DOS Shell** for the first time, copy the file COMMAND.PRG to your own formatted disk. Refer to your *Atari* owner's manual for instructions on copying files from one disk to another. (After you've installed **DOS Shell**, you'll never again have to restart a copy if you release the mouse button too soon.) Put HIGH.PRG in the auto folder to force the color monitor into medium resolution.

After turning on your *Atari ST*, insert the COMMAND.PRG copy into a disk drive. Double click on the appropriate drive icon to get a directory if necessary. Move the pointer to the COMMAND.PRG icon or label and double click to run the program.

> *Note: When programs are placed in the auto folder of the boot disk they are executed in the order they were copied. The **DOS Shell** has a run command that allows you to "RUN" Gem programs. This command will not work if **DOS Shell** was started from the auto folder. The reason for this is that the GEM DESKTOP has not been started at this time.*

The screen will clear, then display an introductory message. You will also see the command prompt with a flashing cursor (unless you change it, the prompt will be the letter of the default disk drive followed by a right bracket: A>.). This is the DOS request for orders. You may type in any of the **DOS Shell** commands described in this manual.

It's called a shell because the **DOS Shell** commands reside outside the GEM system. When you call for a **DOS Shell** command, you temporarily leave GEM control. When you exit the shell, you'll be returned to the GEM system. The DOS Shell exit routine notifies GEM of all modifications you made. It will not be possible to tell later, for instance, whether you used GEM or **DOS Shell** to copy a file.

Chapter 14:

Structuring Your Files

Structuring Your Files

Whenever you format a new diskette, the operating system lays out a physical structure and a logical structure for the way data will be stored on the diskette. As an operator, you only need to deal with the logical structure. The operating system takes care of the relationship between the logical and the physical.

For the simplest file handling you might never need to know how the operating system stores files on your disks: after all, the operating system is supposed to do the housekeeping for you and let you work on more important things.

In fact, if you are willing to organize your files simply as a list on each disk, you need not read the rest of this section. You will need to learn only the **Basic Operations** and **File Manipulations** commands from DOS Shell. You should resume reading at the **DOS Shell Command Style** section.

But if you want to benefit from a structured data storage method, you'll need to understand how directories and subdirectories work.

The Root Directory

Near the outside edge of each disk, a few sectors of the disk are reserved for a list of all files assigned to the root directory on the disk. This directory saves for each file the name, the date and time of creation, the attributes, the first sector and cluster used, and the size. The number of sectors used for the root directory depends on the type of disk.

The number of files that can be stored on a disk may be limited by the number that can be stored in the root directory, even though there may be empty storage space available on the disk. This is a consequence of the limited space originally assigned to the root directory. The system won't let you create another file if it can't write down where to put it.

One reason for creating subdirectories, then, is to expand the number of files you can store on one disk. Because the subdirectories themselves are files, there is no preassigned length restriction for the subdirectories, and so no restriction on how many files may be assigned to one (except, of course, for completely filling the disk space).

Subdirectory Sequences

Subdirectories (the functional equivalent of folders in GEM terminology) are created in a "tree" sequence. Using such a structure imposes order on the logical structure of the information on the disk and lets you keep track of files more easily.

The main or root directory (what *Atari* or GEM would call a base folder) is the one the system always starts with on any diskette. Physically, it is the one stored in the reserved area near the outside edge of the diskette.

Within the main directory you can create a subdirectory (or folder as *Atari* or GEM would call it) by using the **DOS Shell** command *Make Directory*. Within this subdirectory, you can create another subdirectory. In fact, you may create as many subdirectories as you like.

Each subdirectory belongs to (or grows from, if you like the tree analogy) the directory or subdirectory it was created in. The first level of subdirectories belong to the root directory. The second level directories each belong to a particular one of the first level directories, which in turn belongs to the root directory.

To find or create a file in the second level directory, you would usually have to tell the system not only the name of the file, but also the names of the second level directory and the first level directory.

You may stack subdirectories as deep as you like. The only limits are that there may be no more than the maximum number of files, counting each subdirectory as a file, in the root directory and that there be enough disk space for all the subdirectories and files.

Manipulating Subdirectory Files

Once you've created a subdirectory, you may assign files to it by an appropriate *Save* or *Copy* operation in which you specify the path, or sequence of directory and subdirectories in the logical chain leading to the file. (There are defaults and shortcuts to reduce the typing load.)

The path operates as a road map for the computer so that it can figure out where to look for the file. Generally, you will have to tell the computer to begin with the root directory, then proceed through a specific chain of subdirectories, each branching from the previous subdirectory. When you reach the subdirectory that contains the file you're interested in, you just give the filename of the file.

If the appropriate subdirectory belongs to a chain of subdirectories that includes the current directory as a parent, you may simply specify the path from the current directory instead of from the root.

You may delete files from a subdirectory, again by specifying both the file name and its associated path. If a subdirectory is no longer needed, you may delete it as you would any other file, except that you must first delete all files assigned to it. That is, you may remove a subdirectory, but only if it's empty.

Default Drive

Whenever more than one drive is available, **DOS Shell** assumes that one of them is the default drive. The system will try to use the default drive to carry out any command you issue without specifying a drive.

When **DOS Shell** is first run, it shows a prompt such as A> or B>. The A> shows Drive A is now the default drive.

If you wanted to see a list of files assigned to the default directory, you could type the DIR (*Show Directory*) command without a drive or directory parameter. The system would then examine the disk in the default drive and print the default directory it finds there.

To Change the Default Drive:

At the command prompt, type a drive letter followed by a colon. For example, typing b: would change the default drive to Drive B. The default drive remains the same until you order a change.

Default Directory

DOS Shell also assumes a default directory on each disk. This is the directory the system will use whenever you issue a command without specifying a path.

Whenever you turn the computer on, your default directory will be the same as would be opened on the Desk Top of the default disk.

To Change the Default Directory:

Use the *Change Directory* command. Once assigned, the default directory remains the same until you order a change.

Chapter 15:

DOS Shell Command Style

DOS Shell Command Style

Command Format

Every **DOS Shell** command is a string of words and symbols. In the descriptions of the commands, you'll see such strings of words. It's important to know what role each plays, whether it's a special word or a stand-in, and whether it's required or optional.

The command line consists of a command keyword or the name of a program to be executed, and often some other text on which the command operates. Each command ends with RETURN.

A typical command, for instance, is

DEL file05

Here DEL is the command and file05 tells the system which file to erase.

If you just want to run a program, you simply type its name and press RETURN.

The command to run the program *Foldit* would be *Foldit*.

If *Foldit* requires the GEM Desktop environment to run, you'll need the special command RUN described in this manual.

If the program or file is not on the current directory on the default drive, you will have to tell the system where to look by specifying the path.

The MICROSOFT, INC.™ style was adopted to distinguish the rules for the various words and symbols.

The words shown here in UPPERCASE are the commands; they must be typed character for character as shown. They are keywords in the same way that BASIC commands are keywords.

There is one option available: you may actually type these words in either upper or lower case; **DOS Shell** ignores the case.

Following each command or keyword may be more text, usually called the parameters of the command. Not all of this information may be required every time.

If the information is printed in italics, the words in this manual are dummies, stand-ins for the names of files or other objects appropriate in a given situation. You substitute the right names each time you issue the command.

If the information is enclosed within [square brackets], it is optional for at least some purposes. For instance, if you want to list the files in your current directory (the DOS name for a folder), use the command:

DIR

If you want to see the files on disk Drive B without making Drive B the current drive, type:

DIR b:

Here b: is a parameter of the command DIR. It's optional in the sense that the command works (does something) without it, but you need it frequently.

Any parameters not enclosed within square brackets must be included with the command.

For example, when you copy a file to another disk, you must give at least one file name so **DOS Shell** will know which file you want copied.

An ellipsis (...) means that you may give more than one parameter of this type. For instance, you may type several files in one command by listing them all.

You must include all punctuation except the square brackets as it is shown in this manual, except that you may substitute delimiters as discussed in the next paragraph. Required punctuation comprises the spaces, commas, colons and equal signs that appear in this manual's command descriptions.

Commands and parameters must be separated by delimiters (space or tab). You may use different delimiters within one command. Only spaces have been used in this manual, but you may use tabs if you prefer.

Do not use any delimiters within a complete file specification (d:file name.ext). The : and . already serve as special delimiters in this case.

Command Parameters

The following list describes those parameters you will see when reading the detailed descriptions of the commands. Some commands require no parameters, while others may use all of them. Any special default values of these parameters will be discussed with the commands.

d: This parameter stands for a disk drive letter that tells **DOS Shell** which drive you wish to use. You must enter the colon after the letter. If you omit this parameter when it is optional, **DOS Shell** always assumes you want to use the default drive.

Path: The path shows how to find a particular subdirectory or file within the overall logical structure of the disk. For example:

\branch\limb\twig: This describes a series of directories or folders. The first back slash "\" tells **DOS Shell** to start at the root directory. You must also separate each subdirectory name with a back slash.

If the initial back slash is not included, **DOS Shell** will look for the directories beginning at the current directory (which may or may not be the root directory).

A path may be no longer than 63 characters long from the root directory to the last subdirectory. This includes all back slashes.

If you place a file name after the path, you must also include a back slash between the final directory name and the file:

\level1\file

This would access the file in the directory *Level1*.

File name: Filenames may be from 1 to 8 characters long and include these characters:

A-Z 0-9 _ (underscore character)

Any invalid character in the name will cause the rest of the characters to be ignored.

.ext: The extension to a file name may be from 1 to 3 characters long. It immediately follows the file name and is preceded by a period. The same characters may be used in the extension:

A-Z 0-9 _ (underscore character)

Whenever a file is created with an extension in its name, you must always include the extension whenever you refer to the file. Otherwise, the system will fail to locate the file.

Exception: All executable files should have one of the extensions PRG, TOS, TTP or BAT. Such executable files may be called by typing the file name without extension.

Command Execution

DOS Shell commands begin execution when you press RETURN.

When **DOS Shell** has done as you commanded, the prompt will reappear to indicate that DOS Shell is ready for another command. If a command could not be performed successfully, one or more error messages will appear before the prompt.

To stop a command in progress, press the CONTROL and C keys simultaneously. To prevent damage to the disk, **DOS Shell** will finish the current task before it stops.

Global Filenames

One of the powerful features of the **DOS Shell** is its ability to recognize global filenames. This lets you refer to more than one file by using one inclusive name.

You may do this by using two special characters: * and ?. These characters are called wildcards, because they stand for any legal character in a file name.

For example,

DIR prog?.txt

would list all of these files:

prog1.txt
prog2.txt
proga.txt
prog_.txt

Why are there two wildcards? The ? replaces only one character; the * replaces as many characters as needed.

The following command would list all files with a .txt extension, no matter what their filenames:

DIR *.txt

To copy all of the files from Drive A to Drive B, you could use:

COPY a:*.* b:

To get a listing of all the files beginning with the letter Z, you could use any of the following:

DIR z*.*
DIR z???????.???
DIR z*.???

Command Line Expansion

To help you avoid trouble from inaccurately typing long strings, and to save time and typing generally, **DOS Shell** lets you use as part of the command line abbreviations you assign yourself for commands, filenames, and pathnames.

The abbreviations are the full functional equivalent of the longer strings they stand for; **DOS Shell** will treat them just as it would their longer equivalents.

In effect, the abbreviations expand the length of the command line. The normal 80-character command line can be expanded to as many as 128 characters.

To set up an abbreviation, use the SET command described in the **Directory Structure Commands** section of this manual. Remember to use quotation marks to enclose strings containing blanks.

To use the abbreviation, just type a percent symbol and the abbreviation in place of the string of characters it replaces.

For example, you could use the *Set* command to make the abbreviation d stand for the string dir a:\ Then you could type just %d to see the root directory of the disk in Drive A.

Chapter 16:

Batch Files

Batch Files

A Batch file lets you automatically run a group of programs or DOS commands simply by typing the name of the Batch file. This is invaluable for common, repeated operations like setting up your computer at the beginning of the day.

You can create one special Batch file, whose name must be AUTOEXEC.BAT, that will run whenever you turn on the computer.

You might, for instance, install such a Batch file on your word processor disk to set up your printer, call up a RAM disk and a print spooler, load the word processor and assign the default drive for the word processor.

You may create as many other Batch files as you like. These will run whenever you call them. All Batch files must have the extension .BAT.

You may call one Batch file from another, up to 5 levels deep. Each *IF* Batch subcommand uses up one level.

You may change directories as often as you like within the Batch file. The system remembers which directory the Batch is located on, so it can return when necessary, no matter where you may have moved in the tree of subdirectories.

Do not remove the disk containing the Batch file from its original disk drive. TOS will have trouble dealing with a different disk.

You may ask the system to:

- Run any program, including **DOS Shell**.
- Carry out any **DOS Shell** command.
- Carry out one of the special Batch subcommands.

Creating Batch Files

To create a Batch file, use any word processor or line editor that produces ASCII code. Descriptions of each **DOS Shell** command and of the special Batch subcommands make up the latter portion of this manual.

Each command or program call should have its own line, up to 80 characters in length and ending with RETURN. The commands are carried out just as if you had typed them in at the keyboard. You may pass parameters when the Batch file executes. See the section on Batch files with replaceable parameters for details.

For example, the following Batch file will set A: as the default drive, send the printer setup file setup from Drive A: to a printer *PRN*, install MICHTRON's **M-Disk** and **Spooler** from Drive A:, and call the program *Mince* from Drive B:.

```
a:
TYPE setup prn
mdisk
sspool
b:mince
```

Once you've chosen and typed in all the commands with your word processor or line editor, save the file to an appropriate disk.

You may use any legal file name you like, but:

- Only the file AUTOEXEC.BAT will run automatically on booting up.
- You must name the file with the extension .BAT.

Build File Name *Command*

A build command has been included to allow you to make small Batch files.

To use this command:

Type: BUILD FILE NAME...

To exit: press CNTRL Z at the beginning of a new line and then press RETURN

Running Batch Files

To make a file run automatically whenever you start your computer, name it AUTOEXEC.BAT, store it in the root directory of a disk and load that disk into the default drive each time you start your computer. (You would also, of course, have to see that the programs AUTOEXEC.BAT calls were available.)

If you name a file with any other file name and the extension .BAT, it will execute any time you call it by typing its name as a **DOS Shell** command. You need not type the extension.

All commands execute sequentially, except as redirected by the special Batch subcommands GOTO and IF.

You may halt execution by pressing the CONTROL and C keys at the same time. The system will finish the task now executing. When it next tries to read a **DOS Shell** command it will quit.

Creating Batch Files With Replaceable Parameters

You may create a Batch file with up to 10 different dummy parameters you can substitute for at execution time. (See the Batch subcommand SHIFT for the way to use more than 10 variables.) This can be handy when you want to do a standard sequence of operations on different files.

Type the normal sequence of commands and programs, but instead of putting in specific names for the variable parts of the commands, use the special codes %0, %1, ...,%9.

The code %0 always stands for the drive and file name of the Batch file itself. The others may be anything you like.

For instance, the following Batch file would copy a file, then list the copy on the video screen:

```
COPY %1 %2
TYPE %2
```

Running Batch Files With Replaceable Parameters

Type the name of the Batch file, then the name you want to use this time for each of the variable parameters. Separate each with a space, and be sure you type all parameters in the same order you listed them in the Batch file. Press RETURN to start execution.

The Batch file above, if we called it copy1, could be called to copy the file *Fred* on the default drive to the file *Ted* on drive b: by typing:

copy1 fred b:ted RETURN

Chapter 17:

DOS Shell Commands

DOS Shell Commands

The **DOS Shell** commands have been divided by function into five groups. Depending on your needs, you may have to read about and learn from two to five of the groups.

The first group, which everyone will need to learn, consists of basic operations: setting date and time, displaying basic information, exiting to TOS.

The second group, which will be the last some users will need, contains the basic file manipulation commands. If you do not need a structure of subdirectories, Batch operations or input/output redirection, you can stop after you've mastered this section.

The third group of commands deals with the subdirectory structure: creating and removing directories, setting paths through subdirectories.

The fourth group of commands comprises the special Batch subcommands. You'll need these only if you use fairly sophisticated Batch files. Simple Batch files don't require these special orders.

The last group of commands lets you redirect standard input and output to different devices. You could have material that normally appears on your screen transferred to the printer, for instance.

The last page of this manual shows an alphabetical list of all the **DOS Shell** commands so that you can quickly refer to the right page when you've forgotten some small point.

Basic Operations

HELP

Type: ?

This command displays a list of all commands.

CLS (Clear Screen)

Type: CLS

This command clears the screen and places a new command prompt onto the screen.

CHKDSK or CK (Check Disk)

Type: CHKDSK [d:]
or
CK [d:]

This command asks the computer to give you an update on disk usage. It prints the number of bytes of disk storage free for your use, the number of bytes available on the disk (determined by the type of disk drive) and the main volume label for the disk.

If you do not specify a disk drive, the system will report on the default drive.

DATE (Enter Date)

Type: DATE [mm-dd-yy]

This command lets you enter or change the current date used by the computer. Unless you have a battery-powered clock circuit, the date resets to the date your version of TOS was issued every time you turn the power on. This date is used every time a file is created or changed, so if you want to keep file histories, it's important to use the right date.

If you enter a valid date after the DATE command, **DOS Shell** will accept and use this date, then display another command prompt. If you just type DATE, the computer will display:

```
Current date is mm-dd-yy
Enter new date:
```

You may then enter a new date (in the form mm-dd-yy or mm/dd/yy) if you want, where:

- mm is a one- or two-digit number specifying the month
- dd is a one- or two-digit number specifying the day
- yy is a two-digit number specifying the year

Note that the 19 in 19yy is assumed by the computer. You do not need to type it in when entering the year.

If you press RETURN when the computer asks for a new date, it will use the current date it reported in the line above the prompt.

Example: This command sets the date to May 12, 1994:

```
DATE 5/12/94
```

TIME (Set Time)

Type: TIME [hh:mm:ss]

This command lets you enter or change the current time used by the computer. Unless you have a battery-powered clock circuit, the time resets every time you turn the power on. The time is used every time a file is created or changed, so if you want to keep file histories, it's important to use the right time.

If you enter a valid time after the TIME command, **DOS Shell** will accept and use this time, then display another command prompt. If you just type TIME, the computer will display:

```
Current time is 12:23:34
Enter new time:
```

You may then enter a new time (in the form hh:mm:ss) if you want, where:

- hh is a one- or two-digit number specifying hours
- mm is a one- or two-digit number specifying minutes
- ss is a one- or two-digit number specifying seconds

If you press RETURN when the computer asks for a new time, it will use the current time it reported in the line above the prompt.

Example: This command sets the time to 5:05pm:

TIME 17:05

VER (Version)

Type: VER

If you type this command, the system will display the current TOS version number of your *Atari* and the current version number of **DOS Shell.**

Example: You will see something like the following when you issue the VER command:

TOS version 0.19 Command version 1.0

VERIFY (Verify Writes)

Type: VERIFY [ON]
or
VERIFY [OFF]

This command lets you choose whether the system verifies that any data written to the disk is correctly recorded.

When the machine is booted, it assumes a VERIFY ON condition and will check to see that data has been correctly recorded. This verification slows down the saving process.

If you type *Verify* without parameters, **DOS Shell** will display the current state (on or off).

PATH

Type: PATH [d:, ...]

This command lets the computer search not only the current disk drive, but also those you list in this command, for any file or program you enter in a subsequent command. It sets a path for the system to follow in looking for any such file.

Unless you have previously used the PATH command, the system will search only the current disk drive for any command or file you refer to. You may extend the search to as many disk drives as you have attached by using a suitable PATH command.

The path remains set until you alter it or leave **DOS Shell.**

This command lets you more easily run groups of programs stored on different diskettes. In more complex data storage schemes involving subdirectories, setting a path is even more important. A more complete discussion of the PATH command is given in the **Directory Structure Commands** section of this manual.

Example:

If your word processor is set up to use Drive B as the document disk drive, you may want to be able to run some other program residing on Drive A from within the word processor.

To do so, you may leave the word processor (however that may work for your program) and issue the PATH command:

PATH a:

any time before you make the program call. Then type:

program

RUN

Type: RUN [path]program name

This command lets you run a GEM application program from the **DOS Shell**, then return to **DOS Shell** for another command. The program will run normally, just as if it were called directly from GEM.

There are two potential sources of trouble, though, that may keep some programs from running.

First, very large programs may fail to run because both **DOS Shell** and TOS use up memory space. Less memory is available to applications than would be the case if either were present alone.

Second, programs with Resource files not included in the current directory with the program may fail because the Resource file cannot be found.

EXIT

Type: EXIT

You will leave **DOS Shell** and return to GEM at the place you left when you called **DOS Shell**.

Chapter 18:

File Manipulation Commands

File Manipulation Commands

DIR (Show Directory)

The description of DIR found here is limited to its use with disks using no subdirectories. For information on its use with subdirectories, see the description under **Directory Structure Commands**.

Type: DIR [-P][-W][[d:][file name][.ext] ...]

When you issue this command, the system will list all the files (or a portion of the files if you give a particular or global file name) on the specified disk or disks. If you give no disk, the system will use the current disk. The listing will be headed by the volume name. Each file will be shown with its size and date of last update.

If you use -P, the scrolling list will pause when the screen is full. When you're ready, press any key to continue the list.

If you use -W, the system produces a wide-display listing of only filenames. This puts more files on the screen, but it conceals the size and update time for each.

Examples:

To list all the files on a disk in Drive A, use the following:

DIR a:

This command would produce a display similar to the following:

```
Volume in Drive A is MINE
Directory of A:\

FILE1   PRG  3584  1-20-85    2:15p
TEXT    TXT  8320  9-23-84    3:03p
TEST    ACC  1792  2-16-85    1:43a
3 FILES
```

To see the listing of just one file, type DIR followed by the file name:

```
DIR test.acc
```

This command would list all Write Files on Drive A with whatever extensions they might have:

```
DIR a:write.*
```

To get a list of all .txt files on Drive B, type:

```
DIR b:*.txt
```

See the section **Global Filenames** for more information on the ? and * file name characters.

COPY (Copy Files)

The description of *COPY* found here is limited to its use with disks using no subdirectories. For information on its use with subdirectories, see the description under **Directory Structure Commands**.

Type: COPY [d:]filename[.ext] [d:] [filename [.ext]]

When you issue this command, the system will copy the first file named to the second file named. Note that the second file can be created with this command and that it may reside on the same or another disk.

If the second file is on another disk, you may omit the second filename. The copy will have the same filename as the original, but not the same drive.

If you copy the first file onto the same disk, you must give a second filename different from the first. The system will not allow you to copy a file over itself. This prohibition is a safety feature.

COPY does not work for devices, as opposed to files. Use TYPE to transfer data to or from a device.

Examples:

Copying with the Same Filename

This example would copy the file *Sample.txt* from Drive B to Drive A without changing the name:

```
COPY b:sample.txt a:
```

The next example would copy the file picture.prg from the default drive (since no drive letter is specified) to Drive B:

```
COPY picture.prg b:
```

To copy all .prg files from one drive to another, follow this example:

```
COPY a:*.prg b:
```

Review the **Global Filenames** section for a description of how to use the * or ? in the filename.

Copying with a Different Filename

In the next example, the file same.txt on Drive A will be copied to differ.txt on Drive B:

```
COPY a:same.txt b:differ.txt
```

Here the data in one.txt will be copied onto the same disk under the name two.txt:

```
COPY one.txt two.txt
```

DEL (Delete Files)

The description of DEL found here is limited to its use with disks using no subdirectories. For information on its use with subdirectories, see the description under **Directory Structure Commands**.

Type: DEL [d:][filename[.ext]]
or
ERASE [d:][filename[.ext]]

When you issue this command, the system deletes the specified file or files from the designated drive. If no drive is given, the file will be deleted from the current drive.

DEL and ERASE perform identical operations. Both are supplied because each name is widely used in other systems.

You can use the global characters ? and * within your filenames to delete many files in just a few keystrokes. When you do that, it's important that you exercise some caution. You won't be pleased if you accidentally erase several month's work by using a wildcard character to create a general name that turns out to include most of your files. See the section **Global Filenames** for the proper use of these characters.

Examples:

The following example will erase the program *Done.now* from Drive B:

DEL b:done.now

To erase all of the files from the disk in Drive B, you would use the following:

DEL b:*.*

To prevent accidental erasure of an entire disk, you'll get a message when you use wildcards in the target file name for a *Delete*.

The message will list the first filename that matches the global pattern and ask whether you want to delete that file.

The message will look like this:

Delete [filename]? Y/N/G/Q

You have four choices:

Yes: Delete this file and show the next file that matches the global filename.

No: Do not delete this file, but show the next file that matches the global filename.

Global: Delete this file and all other files that match the global, except those already considered. Any file you have not already made a decision about will be erased, along with the current file.

Quit: Do not delete this file, and do not show any more filenames. No more erasures will be done.

RENAME or REN (Rename File)

The description of RENAME found here is limited to its use with disks using no subdirectories. For information on its use with subdirectories, see the description under **Directory Structure Commands**.

Type: RENAME [d:]filename[.ext] filename[.ext]
or
REN [d:]filename[.ext] filename[.ext]

This command lets you change the name and extension of the first file listed to the name and extension listed afterward.

You need a drive specification only for the original name of the file. **DOS Shell** will keep the file on the same disk when it is renamed.

Example:

This command would change the name of file *Home.aaa* to *Record.a*:

RENAME b:home.aaa record.a

TYPE (List File)

The description of TYPE found here is limited to its use with disks using no subdirectories. For information on its use with subdirectories, see the description under Directory structure commands.

Type: TYPE [[d:]filename[.ext] ...]

When you type this command, you will see a list of the file's contents on the screen.

Only text files should be addressed with this command. Programs and other files will not print intelligibly. Some word processor files may not display legibly either; it depends on the method used to store text on the disk.

Data will be unformatted except for tab character codes in the text. When a tab code is found, the display will be shifted to the next tab column (8, 16, 24, etc.).

Wildcards can be used with *Type*. This is particularly useful if you want to concatenate files. Just use the redirect function to send all the text to the new, concatenated file.

Example:

This command would list *Letter.txt* located in Drive B:

```
TYPE b:letter.txt
```

Chapter 19:

Directory Structure Commands

Directory Structure Commands

DIR (Show Directory)

Type: DIR [-P][-W][[d:][path][filename][.ext] ...]

When you issue this command, the system will list all the files (or a portion of the files if you give a special filename) in the indicated directory. If you give no path, the system will use the current directory. The listing will be headed by the volume name. Each file will be shown with its size and date of last update. Any subdirectories will be shown with *DIR* in the file size field.

If you use -P, the scrolling list will pause when the screen is full. When you're ready, press any key to continue the list.

If you use -W, the system produces a wide-display listing of only filenames and subdirectories. This puts more files on the screen, but it conceals the size and update time for each.

Examples:

To list all the files in the main directory of a disk in Drive A, use the following:

```
DIR a:
```

For a listing of all files in the subdirectory *Branch1* on the default drive, use:

DIR \branch1

This command would produce a display similar to the following:

Volume in Drive A is MINE

Directory of A:\branch1

```
.                    <DIR>9-01-87
..                   <DIR>9-01-87
FILE1 PRG   13400  1-20-85      2:15p
TEXT TXT    18600  9-23-84      3:03p
TEST ACC    34034  7-16-85      1:43a
5 FILES
```

Note the two special directories denoted by a single or a double period. The single period directory signifies the current directory and the double period indicates the directory immediately preceding this directory.

To see the files on the preceding directory, you would type the following:

DIR ..

To see the listing of just one file, type DIR followed by the file name:

DIR test.acc

This command would list all Write files with whatever extensions they might have:

DIR a:write.*

See the section **Global Filenames** for more information on the ? and * filename characters.

CHDIR or CD (Change Directory)

Type: CHDIR [[d:]path]
or
CD [[d:]path]

This command lets you change your default directory to another. The new directory may be on the same or on a different disk drive.

You may also use this command without parameters to see the current directory path. If you use the command with only a drive letter, you will see the directory path for the current directory of that drive.

The current directory is the first one the system looks through to find files or commands unless a specific directory has been named. Unless you have asked the system to look elsewhere by using the *PATH* command, the system will look only in the current directory. If no drive is specified, the system looks on the default drive.

Examples:

This example will change the directory to the following path or sequence of folders root one two:

CD \one\two

Note that the first back slash caused the path to start at the root directory.

If you issued the following command after you had typed the preceding example, the directory path would become root one two three:

CD three

MKDIR or MD (Make Directory)

Type: MKDIR [d:]path
or
MD [d:]path

When you issue this command, the system creates a subdirectory on the designated diskette.

If you do not specify a drive, the system uses the default drive.

Examples:

The following command would create the subdirectory *Level1* on the root directory:

MKDIR \level1

Note that the back slash causes the system to attach the *Level1* subdirectory to the root directory.

To create another subdirectory within the *Level1* subdirectory, use this command:

```
MD \level1\level2
```

This command will not work if the *Level1* subdirectory was not already on the disk. The system cannot create two subdirectories at the same time. Another method for creating a level 2 subdirectory would be:

```
CD \level1
MD level2
```

The first command changes the current directory to level 1; the second then creates the level 2 subdirectory. Note the missing back slash when level 2 is created. Since no back slash is present, **DOS Shell** assumes that the subdirectory should be made on the current directory.

You may create as many subdirectories as you wish, as long as disk space is available.

There are just two restrictions: the path name for any subdirectory, beginning at the root directory, may not be more than 63 characters long; there may be no more than a set number of files (the number depends on the storage medium) in the root directory.

You may divide the limited number of files on the root directory between subdirectories and files as you like. No numerical restriction applies to subdirectories.

Because back slashes count as characters, the path \level1\level2 has 14 characters.

RMDIR or RD (Remove Directory)

Type: RMDIR [d:]path
or
RD [d:]path

This command lets you remove a subdirectory from a disk in the default or a designated drive.

The directory being removed must be empty (except for the special "." and ".." directory entries) before it may be removed. This command removes only the last subdirectory name in the path; no other subdirectories are affected. You may not delete the current or the root directories, even if they are empty.

Example:

To remove the limb directory from the branch subdirectory, use this command:

```
RMDIR b:\branch\limb
```

The only effect this command has on the branch subdirectory is the removal of the limb *DIR* entry on the branch directory listing.

PATH (Set Search Directory)

Type: PATH [[d:]path[[;[d:]path]...]]
or
PATH ;

The *PATH* command lets you instruct **DOS Shell** to search other directories than the current one for commands or files you request.

If you type PATH; (with only a semicolon) the system will reset the search path to null and **DOS Shell** will search only the current or the specified directory for a command or file. When **DOS Shell** is first called, it assumes the null search path.

If you type PATH by itself, **DOS Shell** will display the search paths (if any) currently being used.

This command is especially useful if you wish to organize all related files in separate subdirectories. You first list a path which contains those subdirectories you will need to access. When you wish to use some file, DOS Shell will automatically search through the specified subdirectories and find it for you.

A common use, for instance, is to organize your word processor program files into a separate subdirectory. After you specify a path to that subdirectory, you can work on your letters or novel from an appropriate data subdirectory.

Invalid parameters in the *PATH* command, such as bad drive letters, will be detected only when the directories actually need to be searched. No initial error check is done.

If a specified path does not exist on the disk when **DOS Shell** is searching, that path will be ignored and the next path used.

Example:

Suppose you enter the following command:

PATH \level1;b:\tree\limb;a:\mylist

When you now type a command, the computer will first search the current directory on the default drive or the directory and drive specified in the command. If it cannot find the command or file there, it then searches \level1 on the default drive. If it still has not found the command or file, **DOS Shell** tries the path b:\tree\limb. If the search is still unsuccessful, the system searches the subdirectory a:\mylist.

If, for instance, you wished to run a program called paint, the following series of commands would perform the search in the same order as specified in the *PATH* command above:

paint
\level1\paint
b:\tree\limb\paint
a:\mylist\paint

If you used this series of commands, the system would return a File Not Found message after each line until the program paint was found; if you used the *PATH* command, such a message would be printed only after all the directories had been searched.

Within each directory, the system searches for paint with these extensions, in order: PRG, TOS, TTP and BAT. All executable programs should have one of these extensions.

TREE (Tree Structure)

Type: TREE

This command shows on your video screen a list of all subdirectory paths. Each individual path is traced from the root directory to its end at the last subdirectory in the path.

Each line displayed represents one complete path. When a subdirectory has several daughter subdirectories branching from it, the first part of the path, even though it is identical for all such paths, will be shown for each.

VOL (Show Volume)

Type: VOL [d:]

When you issue this command, the system shows the volume name for the specified or default disk.

Example:

The *VOL* command would display:

Volume in Drive A is LETTERS

COPY (Copy Files)

Type:
COPY [d:][path]filename[.ext] [d:][path][filename[.ext]]

When you issue this command, the system will copy the first file named to the second file named. Note that the second file can be created with this command and that it may reside on the same or another disk.

If the second file is on another disk or even on a different subdirectory of the same disk, you may omit the second filename. The copy will have the same filename as the original, but not the same drive or path.

If you copy the first file onto the same subdirectory on the same disk, you must give a second filename different from the first. The system will not allow you to copy a file over itself. This prohibition is a safety feature.

COPY does not work for devices, as opposed to files. Use TYPE to transfer data to or from a device.

Examples:

Copying with the Same Filename

This example would copy the file *Sample.txt* from disk Drive B to disk Drive A without changing the name:

```
COPY b:sample.txt a:
```

The next example would copy the file picture.prg from the default drive (since no drive letter is specified) to Drive B's subdirectory (of the root directory) folder1:

```
COPY picture.prg b:\folder1
```

If the disk in Drive B did not have a subdirectory named folder1, **DOS Shell** would assume that you meant to specify a file named *Folder1*. It would create that file and copy *Picture.prg* to it.

To copy all .PRG files from one drive to another, follow this example:

```
COPY a:*.prg b:
```

Review the section **Global Filenames** for a description of how to use the * or ? in the filename.

Copying with a Different Filename

In the next example, the file *Same.txt* on Drive A will be copied to *Differ.txt* on Drive B:

```
COPY a:same.txt b:differ.txt
```

Here the data in one.txt will be copied onto the same disk under the name two.txt:

```
COPY one.txt two.txt
```

This example would copy the file show.prg into the subdirectory *Level1* and call the duplicate file display.prg:

```
COPY show.prg \level1\display.prg
```

DEL (Delete Files)

Type: DEL [d:][path][filename[.ext]]
or
ERASE [d:][path][filename[.ext]]

When you issue this command, the system deletes the specified file or files from the designated drive and directory. If no drive or directory is given, the file will be deleted from the current drive and directory.

DEL and *ERASE* perform identical operations. Both are supplied because each name is widely used in other systems.

You can use the global characters ? and * within your filenames to delete many files in just a few keystrokes. When you do that, it's important that you exercise some caution. You won't be pleased if you accidentally erase several month's work by using a wildcard character to create a general name that turns out to include most of your files. See the section **Global Filenames** for the proper use of these characters.

Examples:

The following example will erase the program done.now from Drive B:

```
DEL b:done.now
```

This example will delete over.txt from the subdirectory labeled *Level2*:

```
ERASE \level1\level2\over.txt
```

To erase all of the files from the current directory, you would use the following:

```
DEL *.*
```

DEL cannot erase a subdirectory. You must use the *RMDIR* command described in above to get rid of an unwanted subdirectory.

To prevent accidental erasure of an entire disk, you'll get a message when you use wildcards in the target filename for a delete.

The message will list the first filename that matches the global pattern and ask whether you want to delete that file. The message will look like this:

```
Delete filename?  Y/N/G/Q
```

You have four choices:

Yes: Deletes this file and shows the next file that matches the global filename.

No: Does not delete this file, but shows the next file that matches the global filename.

Global: Deletes this file and all other files that match the global, except those already considered. Any file you have not already made a decision about will be erased, along with the current file.

Quit: Does not delete this file, and does not show any more filenames. No more erasures will be made.

RENAME or REN (Rename File)

Type:

RENAME [d:][path]filename[.ext] [path]filename[.ext]

or

REN [d:][path]filename[.ext] [path]filename[.ext]

This command lets you change the name and extension of the first file listed to the name and extension listed afterward.

You need a drive specification and path name only for the original name of the file. **DOS Shell** will normally keep the file on the same disk and directory when it is renamed.

If you want to move the renamed file to a different directory on the same disk, you may do so by specifying a suitable path for the second filename.

If you do not specify a path for the second filename, it will be assigned to the current directory.

Example:

This command would change the name of file home.aaa to record.a and assign record.a to the current directory (whether the current directory is \level1 or anything else):

RENAME b:\level1\home.aaa record.a

TYPE (List File)

Type: TYPE [[d:][path]filename[.ext] ...]

When you type this command, you will see a list of the file's contents on the screen.

Only text files should be addressed with this command. Program and other files will not print intelligibly. Some word processor files may not display legibly either; it depends on the method used to store text on the disk.

Data will be unformatted except for tab character codes in the text. When a tab code is found, the display will be shifted to the next tab column (8, 16, 24, etc.).

TYPE supports wildcards. This is particularly useful if you want to concatenate files. Just use the redirect function to send all the text to the new, concatenated file.

Examples:

This command would list letter.txt located in drive b:

TYPE b:*level*1\letter.txt

This command would concatenate the text files text1.txt and text2.txt on the file newtext.txt:

TYPE b:\dir6\text1.txt a:\dir5\text2.txt >> a:\newtext.txt

See the Redirection symbols section for information on using the >> command.

PROMPT (Set System Prompt)

Type: PROMPT [prompttext]

This command lets you change the command line prompt to a different string of characters available from the keyboard.

When waiting for a command, **DOS Shell** will always display a command prompt line. Unless you use this command to change it, a standard prompt will be used.

Besides the normal keyboard characters, the following special characters are available to print information or

special control characters. You type these in the form $c. $c may be any of the following:

$$	Prints a $ character.
$t	Prints the time.
$d	Prints the date.
$p	Prints the name of the current directory on the default drive.
$v	Prints the version number of the operating system.
$n	Prints the default drive letter.
$g	Prints the > character.
$l	Prints the < character.
$b	Prints the | character.
$q	Prints the = character.
$h	Causes a backspace and erasure of the previous character.
$e	Sends an ESCape code.
$_	Sends a carriage return and line feed (to start printing on the next line).

If you type any other two-character string beginning with $, the system will print a blank space.

The prompt ends at the first space or carriage return typed.

Examples:

To show the directory name surrounded by exclamation marks as the prompt, you could type the following:

```
PROMPT !!$p!!
```

In this example, a two-line prompt would be created:

```
PROMPT Time=$t$_Date=$d
```

It would display the following:

Time = Current time
Date = Current date

To start a prompt with one or more spaces, use a null or invalid character at the beginning of the string. Then place your space or spaces afterward. For example:

PROMPT AAHello$_

would place two spaces before the *Hello* prompt.

Alternatively, you may use quotation marks to create strings containing spaces. You could produce the same prompt as above with

PROMPT " Hello$_"

SET (Set Environment)

Type: SET [name=[parameter]]

This command creates a list containing strings and their identifying names. Any program may then use the names.

The strings are stored in a reserved 1-kilobyte block of memory. The entire string, beginning with its name, is stored into this memory by lines. Names are all converted into uppercase, while the parameter string is stored in whatever form it was typed. If there are blanks in the string, enclose the string in quotation marks.

If you type the *SET* command without name=parameter, the system will display a list of the current strings. When you type SET name=, the named variable and its associated string will be removed from the list.

PROMPT or *PATH* commands use the same memory as the *SET* command and will be included in the list of variables. For example, if you created both a special PROMPT and PATH (by the appropriate commands), they would be listed as *PROMPT=xxx* and *PATH=xxx*.

The following two commands would have the same effect:

```
SET PROMPT=$p$_
PROMPT $p$_
```

Examples:

The following command would add the string letter=\hello\again to memory:

```
SET letter=\hello\again
```

To remove this string from the list, type:

```
SET letter=
```

Chapter 20:

Batch Operation Commands

Batch Operation Commands

ECHO

Type: ECHO on
or
ECHO off
or
ECHO message

This command lets you choose whether the screen displays your commands as they execute from the Batch file.

Messages from your program or from **DOS Shell** always appear, regardless of what you choose.

When the system is turned on, the echo is also turned on, so the commands will be displayed unless you turn the echo off. If you want to know whether echo is currently on or off, type:

ECHO RETURN.

To write a message from the Batch file, just type it in after the echo command. It will appear on screen when execution of the Batch file reaches the echo command line.

GOTO

Type: GOTO label

This command lets you transfer control to a line other than the one immediately following in the Batch file. Control goes to the line immediately following the label .

You create the label by starting a line in the Batch file with a colon, then typing in the label name you want to use (no space required). End the line with RETURN as usual.

A label may be as long as you like, but the system only reads the first eight characters.

For example, the following set of commands would print a file *Happybd* over and over on the printer *PRN*:

```
:loop
TYPE happybd prn
GOTO loop
```

If you have the echo turned on, you would see the *COPY* and *GOTO* commands appear in rotation on the screen.

IF

Type: IF [NOT] condition command

This command gives you conditional control over the execution of the commands within a Batch file.

You may ask that a command be executed if a condition is true or, by inserting the optional NOT, if it is false.

Any **DOS Shell** command or program call may be used as the command.

Just three condition types may be used:

Error Level Number:

This is true if the exit error code from the program or command just executed is number or higher. You might want to use this conditional to avoid running later programs

that use as input the results of the program that failed to exit normally.

Error codes must be generated by the programs called. Not all programs produce such codes.

String1 == String2:

This is true if the strings are identical. This comparison is not case-sensitive. Note the double equals sign required.

The strings compared may be strings written explicitly into the condition, may be variables you type in as part of the Batch file command, or may be variables filled in as output from programs called by the Batch file.

EXIST [d:][path]filename[.ext]:

This is true if the file named exists on the specified directory and drive.

Three more conditions may be formed by inserting NOT between IF and condition.

For example, we might need to alter the command sequence if file *Harvey* is missing:

```
...
IF NOT EXIST harvey GOTO later
RUN harvfix
TYPE harvey
:later
...
```

PAUSE

Type: PAUSE [remark]

This command lets you suspend the Batch process and display the message remark on your monitor.

You will also get the second message Strike a Key When Ready.... The Batch file will begin to process again with the line following the pause command as soon as you press any key except the CONTROL-C combination.

Remark may be up to 70 characters long.

Typical uses for pause commands are to let you swap disks, change printers or print heads.

REMARK

Type: REM [remark]

This command simply displays the remark on your monitor when execution reaches this line. The remark may be up to 80 characters long.

You might want to put such remarks in your Batch files to remind you what's going on as the Batch file executes.

SHIFT

Type: SHIFT

This command lets you use more than 10 replaceable parameters in a Batch file.

Each time you issue the shift command, the replaceable parameters are renumbered one number lower than they were before.

Parameter %9 becomes %8, for instance. At the same time, an eleventh parameter that previously had no name gets the name %9.

If you issue the shift command again, a twelfth parameter acquires the name %9 and the eleventh becomes %8.

The first two replaceable parameters would no longer be usable after the shift command was issued twice.

You cannot "unshift" to get back parameters shifted off the list of %n names.

Redirecting Input and Output

Sometime, you may want to make your programs execute with data from another source than your keyboard or store their results somewhere other than on your screen. The redirection features described here let you shift either input or output to files or other devices instead.

This can be handy for treating data collected earlier and stored in a file. It's also easy to break down a large data manipulation into smaller programs that feed data to each other. Combining text files is easy when you TYPE a series of files and redirect the output to a file.

Redirection is also convenient for sending data to or from nonstandard input devices. If you want a paper record, just redirect screen output to the printer. If you want to type at a remote console, just redirect keyboard input to that device.

DOS Shell supports the standard devices *PRN* for printer, CON for keyboard and AUX for RS232.

All redirection techniques work only if all programs involved use standard input and output. Nonstandard techniques will cause redirection attempts to fail.

Three basic redirection techniques are available:

- ***Redirection symbols*** route data to or from specific files or devices.

- ***Pipes*** are temporary files that hold the output of one program for a second one to work on. The temporary files are erased as soon as the second program is finished.

- *Filters* read the contents of files or other input streams and write modified results to an output file or device. The original data is not changed.

Redirection Symbols

The *Redirection Symbols* <, > and >> let you reassign standard input or output.

The format is:

SYMBOL[d:][path]filename
or
SYMBOL[d:][path]devicename

< This assigns input that would otherwise come from the keyboard to the file filename or the device devicename. If the file or device does not exist, the system will complain, then try to run the program with the keyboard as the input device.

When devices or files produce input, they use Control Z as their end-of-file mark.

> This creates the file filename and routes output that would otherwise go to the video screen to filename. If filename or devicename already exists, this command will truncate its contents to zero length before assigning it for output.

>> This routes output that would otherwise go to the video screen to the file filename or the device name devicename. The new output will be added to the end of the contents of filename or devicename. If filename does not exist, it will be created. With a freshly-created file, > and >> have the same result.

Examples:

To create the file textlist and load it with the names of the text files on Drive B, type

```
DIR b: >textlist
```

To use the file testdata as input to the program massage, type:

```
massage <testdata
```

Pipes

The pipe symbol | lets you send standard output from one program to standard input to another program.

The pipe creates a temporary file, whose name is of the form PIPExx.IN or PIPExx.OUT on the root directory of the default drive. Standard output from the first program named goes there; the second program named reads from there as its standard input.

The pipe symbol has the effect of using a combination of output and input redirection symbols, except that the file used to store the output is erased at the end of the sequence.

Example:

This command line would take the output from the program glenda and send it to the program *Jackson*:

```
glenda | jackson
```

Filters

Filters let you see on a standard output device a modified version of the (unchanged) data from a standard input device.

The three filters (**More, Sort**, and **Find**) modify the input data in different ways.

More

This sends the standard input to the screen, then pauses when the screen is full. If there's more data to read, the screen shows:

```
More
```

at the bottom. Press any key to see the next screen of data.

For example, to display the root directory of the disk in Drive B one screen at a time, type:

```
DIR b: | MORE
```

To see the contents of the file *test.one* one screen at a time, type:

```
MORE <test.one
```

Sort

This rearranges the standard input data into the ASCII collating sequence and writes the results on a standard output.

The format is:

SORT [-R][-n]

The switch -R reverses the sequence.

The switch -n specifies the column in which the sort begins. The default is column 1.

Files to be sorted must fit in available memory.

For example, the following command would sort the file tend and write the result on the file dent:

SORT <tend >dent

An alphabetically-arranged directory can be recorded on file list with the command:

DIR | SORT >list

Find

This displays on the standard output device all lines from a standard input device that match a specified text string. The format is:

FIND string

For example, this would list all .PRG files on the \test subdirectory on the disk in Drive B.

DIR b:\test | FIND .prg

Index

Index

STUFF

SuperDirectory

M-Disk Plus

DOS Shell

Come and join us at the Roundtable,™ Where the GEnie™ and the Griffin meet!

Does this sound like a fantasy? Well, it may just be a dream come true! When General Electric's high-tech communications network meets MICHTRON's programmers and support crew, ST users around the country will hear more, know more, and save more.

We know that our low prices and superior quality wouldn't mean as much to you without the proper support and service to back them up.

So we are now available on GEnie, the General Electric Network for Information Exchange. GEnie is a computer communications system which lets you use your personal computer, modem, and communication software to gain access to the latest news, product information, electronic mail, games, and MICHTRON's *own* Roundtable (See the special MICRODEAL Section for game information)!!

The Roundtable Special Interest Groups (SIG) gives you a means of conveniently obtaining news about our current products, new releases, and future plans. Messages directly from the authors give you valuable technical support of our products, and the chance to ask questions (usually answered within a single business day).

GEnie differs from other computer communication networks in its incredibly low fees. With GEnie, you don't pay any hidden charges or minimum fees. You pay only for the time you're actually on-line with the MICHTRON product support Roundtable, and the low first-time registration fee.

For more information on GEnie, follow this simple procedure for a free trial run. Then if you like, have ready your VISA, Mastercard or checking account number and you can set up your personal account immediately -- right on-line!

1. Set your modem for half duplex (local echo)--300 or 1200 baud.
2. Dial **1-800-638-8369**. When connected, type **HHH** and press **Return**.
3. At the U#= prompt, type **XJM11957,GENIE** and press **Return**.

And don't forget, MICHTRON's Bulletin Board System, The Griffin BBS, is still going strong (the griffin is the half-lion/half-eagle creature on our logo). Our system is located at MICHTRON headquarters in Pontiac, Michigan. For a trial run, call (313) 332-5452.

United States
576 S. Telegraph
Pontiac, MI 48053
(313) 334-5700

MichTron®

United Kingdom
Box 68 St. Austell
Cornwall, PL25 4YB
0726 68020